MASTER EN SCIENCES INFORMATIQUES

E.A.O. Projet d'aide à la compréhension des opérations arithmétiques pour l'enseignement primaire

Tonneau, Catherine

Award date:
1985

Awarding institution:
Universite de Namur

Facultès Universitaire Notre-Dame de la Paix (Namur)

Institut d'Informatique

E . A . O.

PROJET D'AIDE A LA COMPREHENSION DES OPERATIONS ARITHMETIQUES POUR L'ENSEIGNEMENT PRIMAIRE.

Mèmoire prèsentè par

Catherine TONNEAU

en vue de l'obtention
du titre de
licencié et maître en informatique

Promoteur Claude CHERTON.

Année académique 1984 - 1985

Remerciements.

Avant tout, je tiens à remercier pour sa patience, sa disponibilité et ses conseils, le professeur Cherton qui a accepté la direction de ce mémoire.

Ma gratitude va ensuite à Monsieur Thomas, instituteur à Floreffe, pour sa précieuse collaboration et sans qui ce mémoire n'aurait pas été possible. Je remercie également sa famille pour l'accueil qu'elle m'a reservé.

Merci enfin aux parents et amis qui, de près ou de loin, ont contribué à l'élaboration de ce travail.

TABLE DES MATIERES.

Ninety-ninety Rule of Projet Schedules -

The first ninety percent of the task takes ninety percent of the time , and the last ten percent takes the other ninety percent.

INTRODUCTION

Ce mémoire traite d'un projet d'E.A.O visant à améliorer la compréhension des opérations arithmétiques dans le cas de la structure additive.

Le premier chapitre permettra de situer notre démarche parmi l'éventail des possibilités existantes en E.A.O. . Les chapitres suivants seront consacrés à la présentation du projet au travers des différentes étapes de son cycle de vie : élaborationdu projet, analyse fonctionnelle, analyse organique et implémentation.

Ce travail est destiné à servir à la fois de mémoire de fin d'étude et d'exemple de projet en E.A.O. . Nous souhaitons au lecteur, qu'il soit informaticien ou enseignant, une agréable lecture.

Chapitre 1: ENVIRONNEMENT E.A.O.

Dans ce chapitre, nous présentons le contexte dans lequel nous avons travaillé : le domaine de l'enseignement assisté par ordinateur ou E.A.O.. La première partie traite des principales questions qui gravitent autour de ces trois initiales. Les deux autres parties décrivent successivement les choix que nous avons faits et la démarche suivie pour réaliser ce mémoire.

1. E.A.O en question.

1.1. Informatique et enseignement.

L'informatique concerne des domaines de plus en plus variés et nombreux de l'activité humaine et en particulier l'enseignement. Avant de parler E.A.O., il est intéressant de le replacer dans le contexte plus général d'informatique et enseignement.

L'interaction informatique et enseignement se joue à deux niveaux :

- l'enseignement par l'informatique.
- l'enseignement de l'informatique.

L'enseignement par l'informatique est l'utilisation de l'ordinateur dans diverses fonctions relatives à l'enseignement. Nous distinguons :

- l'E.A.O où l'ordinateur apporte une aide à un niveau pédagogique. Il vient s'ajouter à la panoplie des outils existants : livres, tableaux, rétroprojecteurs, etc ... .
- les services que l'ordinateur peut rendre au niveau de la gestion et de l'administration d'une école. Citons : la comptabilité, l'élaboration des horaires, la gestion des élèves, du personnel, de la bibliothèque, etc... .

L'enseignement de l'informatique est l'enseignement d'une discipline autonome au même titre que les mathématiques ou l'histoire. L'objectif est d'ouvrir les esprits à l'informatique: culture, phénomène de société, mode de pensée et technique.

Ces deux aspects (outil et formation à l'informatique) sont en fait complémentaires. D'une part, l'ordinateur, outil au service de l'école et de toutes les disciplines enseignées, constitue une illustration à petite échelle du champ étendu des applications en informatique. De plus, la manipulation et le contact direct avec l'ordinateur lors de sessions d'E.A.O familiarisent avec la machine et peuvent contribuer à la démystifier. D'autre part, une formation à l'informatique développe la possibilité d'un apprentissage plus autonome et plus conscient avec l'ordinateur.

1.2. E.A.O : Définition.

"Il faut définir pour débuter si l'on ne veut finir par buter. Néanmoins, définition n'est jamais finition : une définition n'aura jamais le dernier mot." (Elgozy).

Nous reprenons ici la définition de H. Bestougeff et J.P. Fargette [BES;82] : "L'E.A.O. est l'utilisation interactive de l'ordinateur comme outil pédagogique au centre d'une relation éducative entre apprenant et enseignant. Au cours d'une session d'E.A.O, un apprenant travaille en mettant en oeuvre un didacticiel." S'il existe un consensus sur cette définition de l'E.A.O., il n'y a pas d'accord unanime sur les différentes applications qu'elle recouvre : souvent l'E.A.O est entendu dans le sens d'une option pédagogique particulière, en général celle de l'enseignement programmé (E.P.O); Nous préférons lui donner un sens plus large (cfr. 1. 3.).

L'ordinateur, introduit au sein de la relation entre enseignant et apprenant, est l'objet d'un double mythe. Certains y voient la disparition à moyen ou long terme du maître, d'autres lui dénient toute valeur pédagogique. Si l'E.A.O. a à la fois ses fanatiques et ses détracteurs, cela tient au fait que cette forme d'aide à l'enseignement n'est pas encore appréhendée dans ses limites et ses qualités. Nous pensons d'une part que l'utilisation de l'ordinateur doit être concurrente à d'autres outils pédagogiques. L'acte pédagogique, quand il se veut global, fait presque toujours appel à plusieurs moyens complémentaires, selon le principe de l'alternance [SCH;81]. D'autre part, si l'E.A.O. s'est vu attitrer des applications qualifiées injustement de "didactiques", ses références sont sur certains points très convenables.

1.3. Types de didacticiels.

Nous avons classé les applications pédagogiques selon l'utilisation qui est faite de l'ordinateur. Sachons qu'il existe d'autres classifications [BES;82] [SCH;80] [PEU;83] [HYP;82] . Toutes ces formes d'utilisation de l'ordinateur comme outil pédagogique varient quant au degré de liberté qu'elles laissent aux enseignants et aux apprenants. Plusieurs des caractéristiques décrites ci-dessous peuvent coexister au sein d'un même programme de formation.

1.3.1. Exercices et tests.

Ces didacticiels présentent à l'élève des exercices qui permettent l'assimilation ou le contrôle des notions étudiées. L'ordinateur interroge, corrige et parfois fournit un diagnostic.

1.3.2. Enseignement programmé (EPO).

Ces didacticiels sont également appelés tutoriels. Une matière est présentée à l'élève en une succession de modules élémentaires suivis d'un contrôle de bonne compréhension. On distingue :

- le tutoriel linéaire : les modules s'enchaînent d'une manière linéaire.

- le tutoriel ramifié : en fonction des réponses de l'élève, de l'état de ses connaissances ou à sa demande, on effectue des retours en arrière ou des branchements vers des modules explicatifs.

1.3.3. Documentation-illustration.

L'ordinateur joue le rôle d'une encyclopédie: Il rend accessibles diverses informations de type textuel, numérique ou graphique contenues dans des banques de données. Il ne se contente pas toujours de restituer, mais il peut également transformer (par exemple, passer de l'expression mathématique d'une fonction à sa représentation graphique).

1.3.4. Simulation et modélisation.

L'ordinateur fait office de laboratoire pour la simulation de phénomènes quand la réalité est trop fine, trop vaste, trop rapide, trop lente, trop coûteuse ou trop dangereuse. La simulation est utile pour des expériences en chimie ou en physique, mais aussi pour l'entrainement dans diverses disciplines (aviation, navigation, ...) ou encore pour la compréhension de phénomènes sociaux ou économiques complexes.

1.3.5. Résolution de problèmes.

Ces didacticiels utilisent la puissance et la rapidité de traitement de l'ordinateur pour classer des grands fichiers, effectuer des calculs complexes ou traiter des données statistiques. L'ordinateur donne ainsi accès à certaines matières inabordées jusqu'alors, faute de moyens.

1.3.6. La programmation.

La programmation fait appel à la créativité, à l'imagination, mais aussi à la rigueur et à la précision. Ici, l'apprenant programme l'ordinateur. La programmation lui permet de construire sa connaissance, d'aller à la découverte de notions nouvelles. La programmation en logo par exemple, peut ouvrir à l'exploration de concepts de géométrie.

1.4. Ordinateur : outil pédagogique.

Nous avons vu les multiples usages de l'ordinateur dans un cadre pédagogique. Certaines de ces utilisations ne diffèrent pas fondamentalement de celles que l'on ferait d'un livre ou de la télévision. Néanmoins, l'ordinateur présente des caractéristiques que les autres outils n'ont pas et qui font sa force :

- l'ordinateur est interactif. Il autorise donc le dialogue. L'ordinateur pose des questions, l'apprenant y répond, et vice versa. Toute réponse de l'apprenant peut donc être immédiatement évaluée. Un autre avantage est la nécessité d'une participation active de l'apprenant pour l'avancement du travail.

- l'ordinateur est capable de stocker, traiter, et restituer de l'information. Ce qui lui permet de s'adapter au rythme et à la connaissance de l'apprenant, et de recueillir des informations sur les résultats de l'élève, son cheminement, ses choix, etc ... qui seront utiles à l'enseignant.

Si l'ordinateur est un outil puissant, en revanche son utilisation nécessite une véritable formation, comme les autres outils pédagogiques. Nous touchons ici au vaste problème de la formation des enseignants :

- qui formera les enseignants ?
- quel sera le contenu de cette formation ?
- sera-t-elle la même pour tous ?
- quels seront les moyens mis en oeuvre ?

En Belgique, à la demande du ministère de l'Education Nationale, il s'est créé en 1983 dans les universités de Wallonie et de Bruxelles, un réseau de centres d'information: les centres O.S.E (Ordinateur au Service de l'Education). Ces centres, par le biais de séances d'information et de stages pratiques, sont chargés de faire connaître aux parents, enseignants et enfants, les possibilités que peut offrir l'introduction de l'ordinateur à l'école. Le travail réalisé dans ces centres est important, mais est loin d'être suffisant. Le relais en matière de formation est assuré par les universités ou les écoles supérieures et certains groupes de volontaires. La formation y est essentiellement technique, parfois doublée d'une réflexion sur l'utilisation de l'ordinateur à l'école. Les coûts de participation sont à charge des enseignants.

1.5. Les enseignements.

L'E.A.O vise l'enseignement aussi bien public que privé, et ce à tous les niveaux, du primaire à l'enseignement supérieur jusqu'à la formation continue.

Si l'on se réfère à l'effort fourni dans les pays voisins, la Belgique est à la traîne. En l'absence de politique et de plan d'ensemble au niveau national, les initiatives proviennent des écoles elles-mêmes qui se fournissent en matériel informatique divers et les expériences très valables en matière de didacticiel qui sont réalisées à titre individuel par des enseignants courageux, restent le plus souvent inconnues. Un effort est fourni par les universités en matière de coordination, de formation et de développement de didacticiels.

1.6. Les auteurs.

Deux agents sont à l'origine de la production des didacticiels : les informaticiens et les enseignants.

Les informaticiens travaillent le plus souvent pour le compte de firmes qui vendent ou louent des didacticiels. Les didacticiels restent rares sur le marché, qui est d'ailleurs essentiellement Anglo-Saxon. Certains de ces produits sont totalement rigides, d'autres peuvent être paramétrés par l'utilisateur selon ses besoins. Ces différents produits ne nécessitent qu'une légère formation en informatique.

Les enseignants souhaitent pouvoir écrire eux-mêmes des programmes adaptés à leur pédagogie, à leurs problèmes. Mais cela nécessite une formation plus complète en informatique et notamment l'apprentissage d'un langage de programmation. Le type de formation variera en fonction de la complexité des problèmes auxquels l'enseignant souhaite s'attaquer. Certains langages (tutor, ego, love, ...), appelés langages d'auteurs, ont été spécialement conçus pour le développement de didacticiels. Mais ces langages sont extrêmement limitatifs puisqu'ils permettent essentiellement de réaliser des didacticiels de type tutoriel. De plus, si les constructeurs proclament que ces langages ont été spécialement conçus pour des non-informaticiens, en réalité leur apprentissage nécessite une véritable formation.

Les informaticiens sont également à l'origine de la création de packages. Ces packages sont des outils qui peuvent être très utiles aux enseignants. Ils permettent en effet de faciliter la construction de leurs propres didacticiels ou peuvent parfois être directement utilisés à des fins pédagogiques (par exemple les packages d'animation graphique).

1.7. Le matériel.

La gamme de matériel s'étend du gros ordinateur central (système PLATO) aux micro-ordinateurs, en passant par toutes les solutions composites. Il n'y a pas de réponse toute faite quant au choix d'une configuration. Il faut tenir compte de plusieurs facteurs :

- un ordinateur pour quoi faire, quel usage lui destine-t-on ?
- de quels moyens dispose-t-on ?

Il ne faut pas non plus négliger les problèmes d'ergonomie scolaire. Des recherches restent à faire dans ce domaine.

Dans les écoles primaires et secondaires d'enseignement traditionnel, les micro-ordinateurs s'imposent nettement par leur prix. Un autre phénomène plaide en leur faveur : la préférence des enseignants pour les petits programmes didactiques portant sur des points précis de matière, qui laissent la possibilité de pratiquer l'E.A.O. " à la carte " et qui ne réclament pas un gros matériel.

2. Les choix.

Nous avons voulu apporter notre concours aux expériences de développement de didacticiels. Il nous a fallu faire des choix concernant le type de didacticiel, les auteurs, l'enseignement et les matériels, parmi le large éventail des possibilités en E.A.O..

Au delà de la réalisation proprement dite du didacticiel, nous désirons que ce travail puisse servir de modèle de "bonne programmation" aux enseignants désireux de réaliser ou de modifier des programmes. Ce travail devrait posséder trois qualités essentielles :

- être bien documenté,
- être aisément modifiable,
- être compréhensible par les enseignants ayant reçu une formation en informatique.

2.1. Les auteurs.

Les enseignants sont les premiers concernés en matière d'E.A.O.. Ils connaissent les problèmes et souhaitent disposer d'outils adaptés à leurs besoins. Cependant, nous l'avons déjà souligné, le développement d'un didacticiel demande du temps et des connaissances en informatique. Devons-nous demander aux enseignants de devenir informaticiens ou aux informaticiens de devenir pédagogues ? Nous pensons que la collaboration informaticien-enseignant est une solution réaliste au problème. Chacun reste le spécialiste de son domaine. Ce travail s'est réalisé avec la collaboration de Mr Thomas, instituteur à l'école primaire de l'Abbaye de Floreffe, responsable de la classe de rattrapage. Ce premier choix a grandement influencé les autres.

2.2. L'enseignement.

L'enseignement visé a donc été tout naturellement l'enseignement primaire. Nous verrons par la suite l'impact que le jeune âge des apprenants a eu sur notre travail.

2.3. Le type de didacticiel.

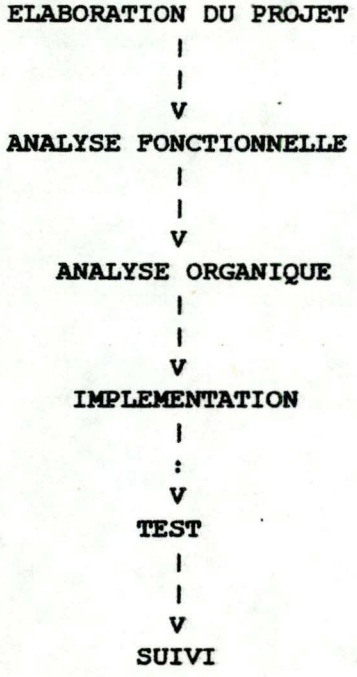
Nous avons choisi une matière enseignée dans laquelle notre didacticiel viendrait s'intégrer : l'arithmétique. A la demande de Mr Thomas, nous avons opté pour la réalisation d'une session d'exercices visant à améliorer la compréhension des opérations d'addition et de soustraction.

2.4. Le matériel.

Nous avons choisi de développer le didacticiel sur micro-ordinateur, car il constitue le support le plus répandu dans les écoles pour des raisons économiques évidentes.

3. La démarche.

La démarche adoptée pour la réalisation de ce didacticiel ne diffère pas fondamentalement de celle du développement d'un autre produit informatique :



L'élaboration du projet et l'analyse fonctionnelle relèvent principalement de la pédagogie, le rôle de l'enseignant y est essentiel. Le suivi est la vérification sur le terrain que le didacticiel remplit bien les objectifs qu'on lui a assignés.

Chapitre 2: ELABORATION DU PROJET

Nous présentons dans ce chapitre une description globale du projet et, au préalable, le cadre pédagogique dans lequel il vient s'inscrire.

1. Environnement pédagogique.

1.1. Les problèmes.

Nous sommes partis de la constatation suivante : certains élèves éprouvent des difficultés à exprimer, à l'aide de l'écriture arithmétique, une situation problématique de vie qui met en oeuvre une situation additive. Et ceci, non parce qu'ils ne maîtrisent pas le calcul, mais parce qu'ils n'ont pas réellement compris les concepts d'addition et de soustraction. Ce problème provient de la difficulté générale des élèves à établir une distinction entre un concept et sa représentation.

Illustrons cette idée simplement : pour enseigner la notion de forme géométrique "carré" à un enfant, on présente généralement le dessin de la figure A. Si vous lui demandez de dessiner un carré, le résultat sera souvent fort semblable à cette figure. Demandez-lui ensuite si la figure B est un carré. S'il maîtrise bien la notion de carré (ou s'il a beaucoup de chance), il répondra "oui". Mais la réponse la plus fréquente est : "C'est un losange.", tout simplement parce que l'enfant a substitué le dessin de la figure A au concept de carré.

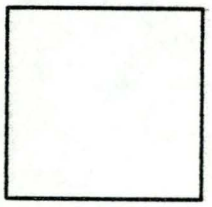


FIG. A

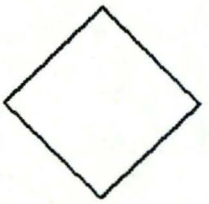


FIG. B

Le processus est similaire pour l'addition ou la soustraction. L'élève travaille avec une représentation (l'écriture arithmétique) qu'il manipule plus ou moins adroitement comme une suite de symboles abstraits : il exerce le réflexe du calcul. Mais face à une situation de la vie qui met en oeuvre la notion d'addition, l'enfant reste démuni pour exprimer et résoudre le problème parce qu'il a oublié le concept caché derrière cette représentation.

Il nous semble donc utile, d'une part, de présenter à l'élève, non pas un seul mode de représentation, mais plusieurs, pour éviter l'identification du concept à la représentation unique qu'on lui en donne, et, d'autre part, de mettre ces différentes représentations en rapport les unes avec les autres pour obliger l'enfant à dépasser le stade de la notation et l'amener à réfléchir et à comprendre le concept commun qu'elles expriment.

1.2. Objectifs-Matière-Méthode.

1.2.1. Les objectifs.

- objectif à long terme :

  - La compréhension du sens des opérations arithmétiques dans le cas de la structure additive.

  Le choix de la structure additive est arbitraire, Nous aurions pu prendre la structure multiplicative.

- objectif à court terme :

  - Etant donnée une situation, exprimée sous une forme donnée, l'enfant devrait être capable de la transposer sous d'autres formes par " reproduction, reconnaissance, ajustement, achèvement de l'initiative, imitation et invention ".

  (tous les termes repris entre les guillemets ont un sens précis en pédagogie. Ils seront explicités au point 1. 2. 3.).

1.2.2. La matière.

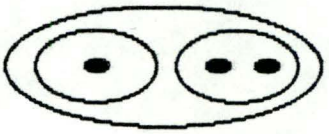
1.2.2.1. Présentation concrète.

- La situation, mettant en oeuvre le concept, matérialisée et jouée.

1.2.2.2. Présentation dessinée ou graphique.

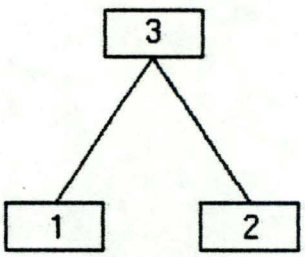
- La situation dessinée avec ou sans animation.

- La représentation en diagramme de Venn.

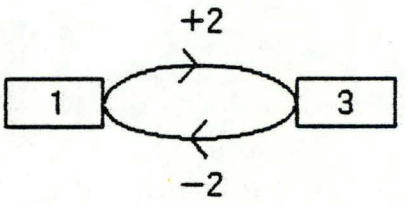


Ces diagrammes constituent un support visuel pour des raisonnements divers; toutefois cette représentation devient lourde lorsque plus de trois ensembles sont représentés.

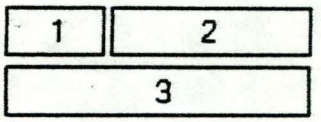
- La représentation arborescente. Elle s'obtient assez facilement par l'observation du diagramme.



- La représentation par graphe sagital, très proche du langage des enfants.



- La représentation linéaire.



- La représentation par écriture arithmétique ordinaire.

Avec l'opérateur plus :

1 + 2 = 3

Avec l'opérateur moins :

3 − 2 = 1

1.2.2.3. Présentation verbale.

- L'expression orale ou écrite du problème.

1.2.3. La méthode.

Nous utilisons la taxonomie de Tirtiaux [HAB;80]. C'est une progression d'activités qui libère progressivement l'initiative des élèves pour les conduire vers des activités d'imitation et d'invention. "La pédagogie de l'imitation est doublée d'une pédagogie de l'initiative dont le but consiste à actualiser efficacement les connaissances fraîchement acquises dans des situations choisies puis imaginées par les élèves eux-mêmes."

1.2.3.1. Reproduire.

Les enfants résolvent individuellement des problèmes analysés en commun dans des situations d'apprentissage ou traitent des cas très proches avec çà et là une variante mineure.

1.2.3.2. Reconnaître.

Dans une série de solutions proposées, les élèves sélectionnent personnellement celle(s) qui correspond(ent) à la notion étudiée.

1.2.3.3. Ajuster.

Les élèves sont en présence d'une situation problématique et d'une solution inadéquate. Ils relèvent l'erreur et la corrigent de manière à mettre situation et solution en parfaite adéquation.

1.2.3.4. Achever l'initiative.

Une situation mettant en oeuvre la notion étudiée est proposée par le maître, les élèves achèvent seuls au gré de leur imagination.

1.2.3.5. Imiter.

Au départ d'un cadre général ou d'un thème global suggéré par le maître, les enfants proposent une ou des situations qui mettent en oeuvre la notion étudiée.

1.2.3.6. Inventer.

Toute initiative est laissée aux élèves qui ont pour seule consigne : "Pensez à ce que vous voulez et imaginez une ou plusieurs situations qui mettent en oeuvre la notion étudiée."

1.2.4. Les prérequis.

Nous pensons que des activités visant l'objectif à court terme peuvent être abordées à partir de la deuxième année primaire. Certains prérequis du niveau de la première année sont nécessaires pour aborder ces activités :

- avoir joué des situations mettant en oeuvre la structure additive;
- comprendre les types de représentations utilisés dans l'activité;
- maîtriser le vocabulaire spécifique : ajouter, mettre ensemble, réunir, réduire etc...

2. Le projet.

2.1. Les besoins.

L'enseignant souhaite disposer d'un outil qui réponde à deux critères :

- le premier est un critère d'intégration : le projet doit s'intégrer dans la démarche pédagogique décrite dans le point précédent et ce, de deux façons :
  - d'une part, il vient couvrir un point bien précis de la matière et non pas toute la matière. Son utilisation peut être concurrente à d'autres moyens, elle reste donc limitée dans le temps.
  - d'autre part, son contenu doit être fidèle aux objectifs pédagogiques que l'on poursuit.
- le second critère est l'individualisation : le didacticiel doit permettre de passer d'un enseignement frontal à un enseignement individualisé. L'enseignement frontal est celui de l'enseignant face à sa classe. Les élèves sont tous attentifs au même moment, ils sont soumis à une même activité, pour une même durée, et au même rythme. L'enseignement individualisé est un enseignement destiné à un élève ou à un petit groupe homogène d'élèves. L'activité, le rythme, la durée sont adaptés à l'élève ou au groupe.

2.2. Objectif-Matière-Méthode.

2.2.1. L'objectif.

- Etant donné une situation exprimée sous une certaine forme, l'élève devrait être capable de la transposer sous d'autres formes par reconnaissance.

2.2.2. La matière.

Nous avions le choix parmi les présentations concrète, graphique, ou verbale.

La présentation concrète est à exclure puisqu'un ordinateur ne permet de travailler que sur des représentations du monde réel. Les représentations orales et par dessins animés ont été rejetées pour des raisons techniques et par manque de moyens. L'expression verbale par écrit a été envisagée plus sérieusement. Deux solutions étaient possibles :

- la génération de phrases syntaxiquement et sémantiquement correctes. Le problème est vaste, et nous n'avons pas voulu nous y attarder.

- le stockage de phrases correctes au niveau syntaxique et sémantique. Le désavantage de la monotonie d'un tel système nous a amené à écarter cette solution.

Nous avons donc sélectionné parmi les représentations graphiques celles qui sont accessibles à un micro-ordinateur:

- le diagramme,
- le graphe,
- l'arbre,
- la représentation linéaire,
- l'écriture arithmétique.

2.2.3. La méthode.

Nous avons choisi de nous situer à l'étape 'reconnaître' de la taxonomie de Tirtiaux. C'est en effet la première étape qui autorise un travail individuel de l'élève. Nous reconnaissons que ce choix est arbitraire, il aurait pu se porter sur les étapes ultérieures 'ajuster', 'achever l'initiative', ou 'imiter'. La réalisation de didacticiels portant sur ces étapes contituerait un prolongement et à la fois un complément à notre travail. Au cours de notre travail, nous essaierons donc de tenir compte de ces extensions possibles, et de fournir des outils qui pourront être récupérés lors de leur développement.

2.3. Le contenu.

Le didacticiel comporte deux parties distinctes. L'une destinée aux élèves : ce sont les exercices qui lui seront présentés, l'autre destinée à l'enseignant : c'est la partie préparation qui devrait lui permettre d'ajuster les exercices au niveau des connaissances de ses élèves.

2.3.1. Les exercices.

2.3.1.1. Exercice 1.

- Etant donné une situation de départ exprimée dans une représentation et trois autres situations exprimées dans une même représentation, sélectionner parmi ces trois situations celle qui correspond à la situation de départ.

Dans cet exercice, deux types de représentations sont mis en présence : la représentation de la situation de départ et la représentation utilisée pour exprimer les trois autres situations. Parmi ces trois situations, il y en aura toujours une et une seule qui correspond à la première, les deux autres comprenant des distorsions.

Exemple :

- Situation de départ : 3 + 4 = ?

- Trois propositions :

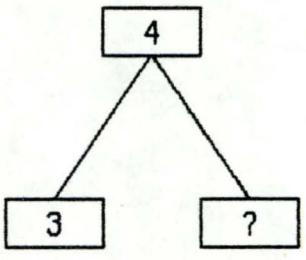


FIG. A

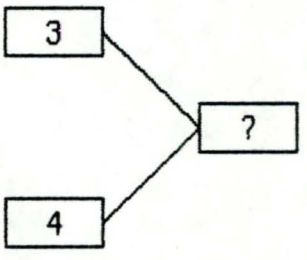


FIG. B

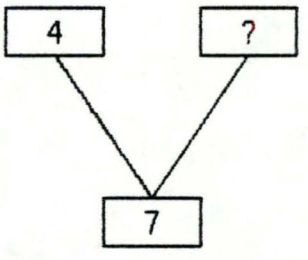


FIG. C

Dans cet exemple, il faut sélectionner la proposition B qui est celle qui correspond à la situation de départ, mais exprimée dans une autre représentation.

2.3.1.2. Exercice 2.

- Etant donné une situation de départ exprimée dans une représentation et trois autres situations exprimées dans trois représentations différentes, sélectionner parmi ces trois situations celle qui correspond à la situation de départ.

Cet exercice est fort semblable au précédent si ce n'est que dans ce cas, les trois situations sont exprimées dans trois représentations différentes. Les types de représentations mis en présence sont plus nombreux. Il y a toujours une et une seule solution.

Exemple :

- Situation de départ : 3 + 4 = ?

- Trois propositions :

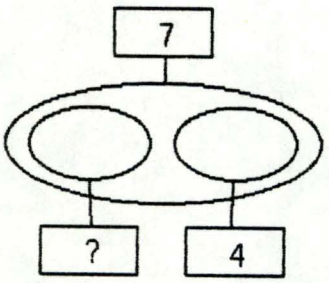


FIG. A.

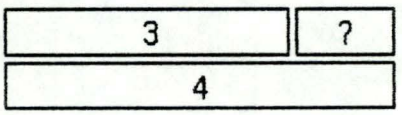


FIG. B

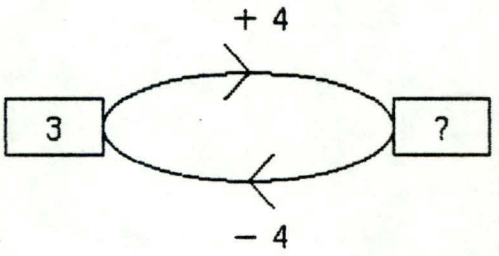


FIG. C

La solution est C.

2.3.1.3. Exercice 3.

- Etant donnè quatre situations exprimèes dans des reprèsentations quelconques, sèlectionner celles qui se correspondent.

Aucune indication n'est donnèe à l'èlève quant au nombre de reprèsentations utilisès, ni quant à l'existence ou au nombre de correspondances entre situations.

Exemple :

- Quatre propositions :

4 − 1 = ?

FIG. A.

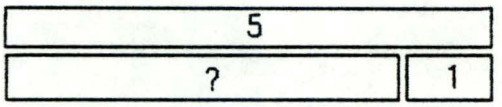


FIG. B.

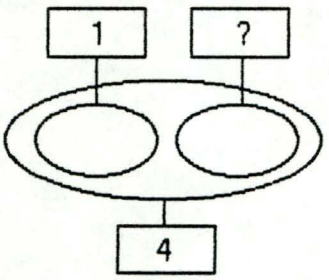


FIG. C..

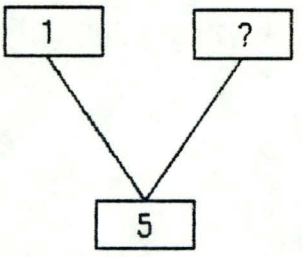


FIG. D.

Solutions : A et C, B et D.

2.3.2. La préparation.

L'enseignant pourra faire un certain nombre de choix concernant chacun des exercices (nombre d'exercices, représentations utilisées, difficulté numérique, ...). Ces choix constituent ce que nous avons appelé une préparation d'un exercice. C'est par ce biais que l'enseignant pourra adapter le didacticiel au niveau de connaissance des élèves. Nous prévoyons que ces préparations soient réutilisables, c'est-à-dire qu'elles puissent être sauvées puis restituées à la demande de l'enseignant. Elles sont accessibles à tout enseignant, informaticien ou non.

Une préparation par défaut est toujours mise à la disposition des enseignants qui, pour une raison quelconque, ne veulent pas entrer dans le processus de préparation. Cette préparation est destinée à des enfants de deuxième année primaire. Elle peut servir de préparation de démonstration du didacticiel.

Chapitre 3: ANALYSE FONCTIONNELLE

1. Introduction.

1.1. Objectif de l'analyse fonctionnelle.

L'objectif de l'analyse fonctionnelle est de fournir une description complète et détaillée des fonctions que le système doit réaliser.

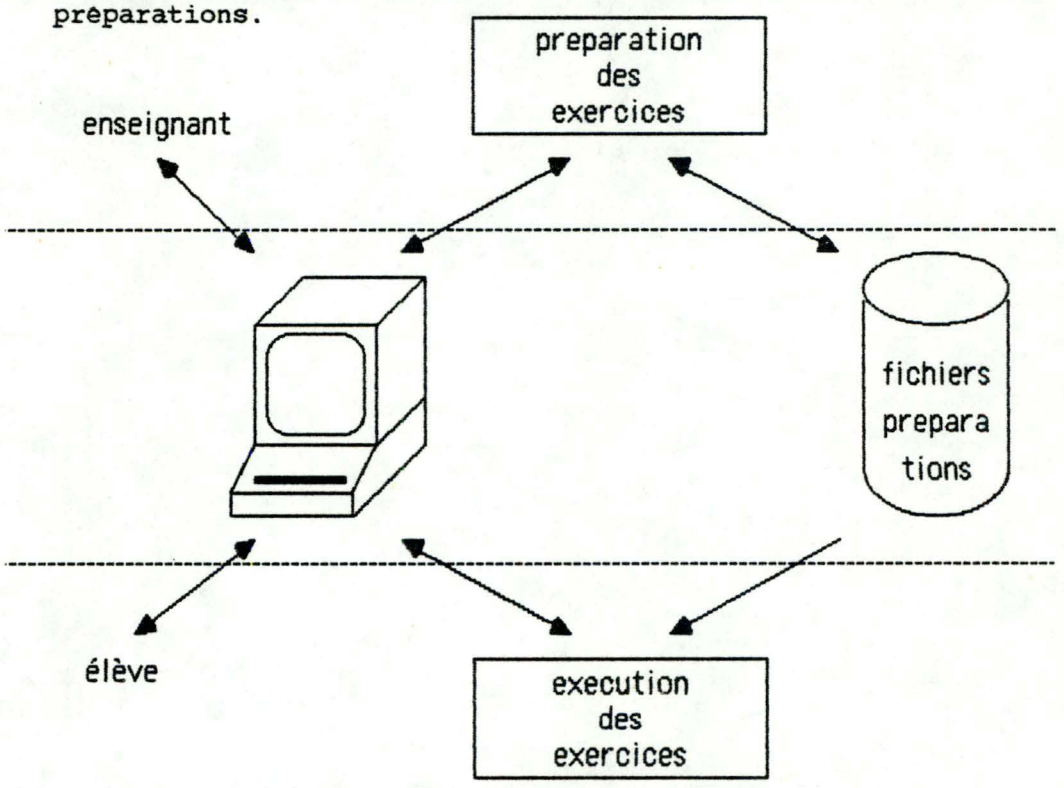
1.2. Démarche.

Nous avons identifié deux fonctions essentielles :

- la fonction de préparation des exercices.
- la fonction d'exécution des exercices.

Ces fonctions se déroulent à des moments différents dans le temps. Elles nécessitent des ressources communes :

- le poste de travail.
- la base de données contenant les informations relatives aux préparations.



Dans ce chapitre on présentera :

- une description des exercices.
- une description de la fonction de préparation.
- une description de la fonction exécution des exercices.
- une description du matériel utilisé.

Un point particulier sera réservé à la notion de maquette.

2. Description des exercices.

2.1. Généralités.

2.1.1. Notion de situation problématique.

Chaque exercice met en oeuvre des situations problématiques de structure additive. Une situation de structure additive met en oeuvre trois entiers positifs, soit A, B et C tels que C est le résultat de la somme de A et B. Par convention, nous appelerons A et B les parties, et C le tout. Dès lors, une situation de structure additive est dite problématique si A, B ou C est inconnu. Donc, on distingue trois formes de situations problématiques :

	A	B	C
1)	connu	connu	inconnu
2)	connu	inconnu	connu
3)	inconnu	connu	connu

Dans la suite, pour éviter de parler de valeur connue et inconnue, nous utilisons la notation suivante pour les trois situations problématiques :

1) A + B = ?
2) A + ? = C
3) ? + B = C

Pour éviter d'introduire des difficultés numériques supplémentaires, A, B et C seront toujours non nuls.

2.1.2. Notion de correspondance.

Pour chaque exercice, l'élève doit établir des correspondances entre les différentes situations qui lui sont présentées. Deux situations problématiques se correspondent :

- si elles mettent en oeuvre les mêmes valeurs numériques pour A, B et C,
- et si la valeur inconnue est la même dans les deux cas.

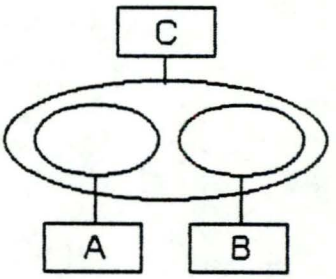
En d'autres termes, une situation S1 ne correspondra pas à une situation S2 :

- si S1 a au moins une valeur numérique différente de S2,
- ou si l'inconnue dans S1 diffère de l'inconnue dans S2.

2.1.3. Notion de représentation.

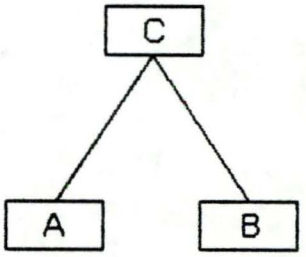
Pour exprimer une situation problématique, on utilise une représentation. Chacune des trois formes d'une situation problématique pourra être exprimée avec l'une des six représentations suivantes :

2.1.3.1. Le diagramme.

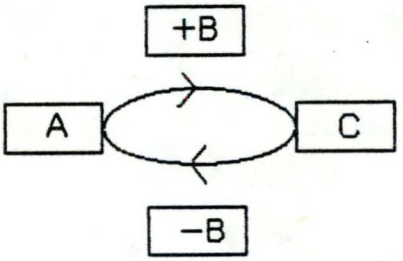


Chaque étiquette associée à un ensemble représente le nombre d'éléments de cet ensemble.

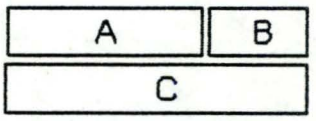
2.1.3.2. L'arbre.



2.1.3.3. Le graphe sagital.

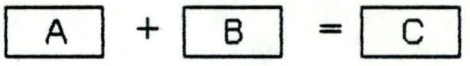


2.1.3.4. La représentation linéaire.

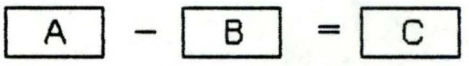


Les longueurs des rectangles doivent refléter les proportions entre les valeurs de A, B et C.

2.1.3.5. L'écriture arithmétique signe +.



2.1.3.6. L'écriture arithmétique signe -.



2.2. Les exercices.

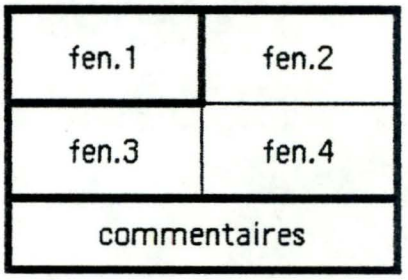
2.2.1. Exercice 1.

2.2.1.1. Enoncé.

Etant donné une situation de départ exprimée dans une certaine représentation, sélectionner parmi les trois situations exprimées dans une même représentation, celle qui correspond à la situation de départ.

2.2.1.2. Présentation de l'exercice.

L'écran est divisé en cinq fenêtres.



La fenêtre 1 contient la situation de départ, elle joue donc un rôle particulier. Pour la distinguer des autres fenêtres, ses contours sont plus épais. Les fenêtres 2,3,4 contiennent les 3 propositions de solution. La fenêtre 5 est destinée aux commentaires et consignes adressés à l'élève.

2.2.1.3. Le dialogue.

Le dialogue élève-machine s'effectue à l'aide du clavier. Les utilisateurs sont de jeunes enfants (7 à 8 ans). Nous avons donc imaginé une solution où la manipulation du clavier reste simple. Dès lors, elle ne devrait pas constituer un obstacle supplémentaire à la résolution des exercices. Nous avons imaginé de dessiner un bonhomme qui peut se déplacer d'une fenêtre à l'autre. Il occupe toujours la même place dans chaque fenêtre, à savoir, le coin inférieur droit. Il est évident que seules les fenêtres 2,3,4 lui sont accessibles. Sélectionner une situation revient à placer le bonhomme dans la fenêtre choisie.

Les trois touches utilisées sont :

- la flèche vers la droite "->" : à chaque pression, le bonhomme se déplace d'une fenêtre, dans le sens des aiguilles d'une montre.

- la flèche vers la gauche "<-" : à chaque pression, le bonhomme se déplace d'une fenêtre, dans le sens inverse des aiguilles d'une montre.

- la touche "RETURN" : c'est la touche de confirmation. Une pression de la touche de confirmation signifie que la situation contenue dans la fenêtre où le bonhomme se trouve est choisie comme solution de l'exercice.

2.2.2. Exercice 2.

2.2.2.1. Enoncé.

Etant donné une situation de départ, exprimée dans une représentation, sélectionner parmi trois situations exprimées dans des représentations différentes, celle qui correspond à la situation de départ.

2.2.2.2. Présentation de l'exercice et dialogue.

La présentation et le dialogue sont les mêmes que pour l'exercice 1.

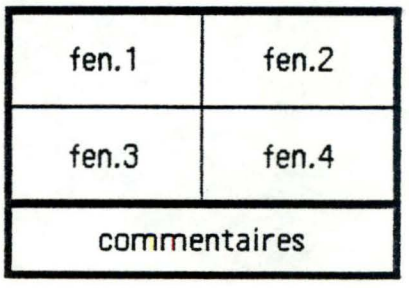
2.2.3. Exercice 3.

2.2.3.1. Enoncé.

Etant donné quatre situations, exprimées dans des représentations quelconques, sélectionner celles qui se correspondent.

2.2.3.2. Présentation de l'exercice.

L'écran est divisé en cinq fenêtres.



Ici, aucune situation n'est privilègiée. Les fenêtres 1,2,3,4 sont identiques et contiennent les représentations des quatre situations. La fenêtre 5 contient les consignes et les commentaires adressés à l'élève.

2.2.3.3. Dialogue.

Le bonhomme peut se déplacer dans les fenêtres 1,2,3,4. Outre la faculté de se déplacer, le bonhomme possède aussi celle de marquer une fenêtre avec un des quatre symboles suivants : "@","#","$","%".

Le but du jeu est évidemment d'assigner le même symbole aux fenêtres qui contiennent des situations qui se correspondent. On considère que les fenêtres non marquées, ou marquées de symboles différents, ne se correspondent pas.

Les touches à utiliser sont :

- la flèche vers la droite "->" , qui garde la même signification.
- la flèche vers la gauche "<-" , qui garde la même signification.
- la touche "@".
- la touche "#".
- la touche "$".
- la touche "%".
- la barre d'espacement, qui permet d'effacer un symbole.
- la touche "RETURN", touche de confirmation.

2.2.4. Remarque.

Pour des raisons esthètiques, mais aussi pour être plus lisibles et plaisants, les exercices devaient être présentés en couleur. La couleur était aussi utilisée pour le marquage des fenêtres de l'exercice 3 à la place des signes (@, %, $, #). En théorie, tout cela est faisable avec un Apple 2 à la seule condition de disposer d'un écran couleur. En pratique, malheureusement, les traits ont la dominante de la couleur souhaitée, mais mélangée à d'autres couleurs. Les caractères subissent le même phénomène, ce qui les rend flous et presque illisibles. Ces effets sont directement liés à des problèmes techniques, mais aussi au fait que nous voulions utiliser la couleur pour le coloriage des traits fins. L'utilisation de la couleur avec Apple 2 peut donner de beaux résultats si on prend une approche de coloriage de surface.

3. Description des préparations.

- Toute préparation se rapporte à un type d'exercice particulier.
- On peut avoir dix préparations maximum par type d'exercice.
- Une préparation par défaut est toujours présente pour chaque type d'exercice.

Pour chaque type d'exercice, il est possible :

- de lister les préparations existantes;
- de créer une préparation;
- de modifier une préparation (seule la préparation par défaut n'est pas modifiable);
- de supprimer une préparation (seule la préparation par défaut ne peut être supprimée);
- d'afficher une préparation.

Une préparation est constituée de choix. Les types choix sont communs aux trois types d'exercices à quelques variantes près. Ils sont examinés dans les points suivants.

3.1. Choix concernant les représentations.

Les figures auxquelles il est fait référence ici sont reprises en annexe.

1. L'enseignant sélectionne, parmi les six représentations, celle(s) avec la(les)quelle(s) il souhaite travailler. Le fait qu'une représentation n'ait pas été étudiée ne sera donc pas un obstacle à l'utilisation du didacticiel. Pour le bon fonctionnement des exercices 1 et 3, un mode de représentation est suffisant, mais l'exercice 2 en exige au moins trois.

2. Pour la représentation en diagramme et en graphe, l'enseignant a le choix entre :

   - un seul dessin (D1 pour le diagramme et G1 pour le graphe),

   - ou plusieurs (D1, D2, D3 et G1, G2).

3. Chaque représentation que l'enseignant aura selectionnée pourra être présentée :

   - soit toujours dans la même position (fig D1, D2, D3, G1, G2, A1, L1, EA+1, EA-1);
   - soit avec des orientations spatiales qui varient (toutes les figures). Les présentations verticales de la représentation linéaire et de l'écriture arithmétique n'ont pas été retenues pour des raisons esthétiques.

3.2. Choix concernant le nombre d'exercices.

L'enseignant ne peut pas associer directement un nombre d'exercices à un type d'exercice. Pour chaque type d'exercice, il sélectionne des sous-types d'exercices auxquels il peut associer un nombre d'exercices. Le nombre total d'exercices qui seront présentés à l'élève est la somme des nombres d'exercices associés à chaque sous-type d'exercice. Les sous-types varient en fonction des types d'exercices.

3.2.1. Exercice 1.

Les sous-types d'exercices de l'exercice 1 sont formés par des couples de représentations (R1,R2), où R1 désigne la représentation qui sera utilisée pour la situation de départ et R2 la représentation utilisée pour les trois autres situations. A chacun des couples de représentations qu'il veut mettre en présence, l'enseignant doit associer un nombre d'exercices (compris entre 0 et 25).

3.2.2. Exercice 2.

Les sous-types d'exercices de l'exercice 2 sont formés par des couples de représentations (R1,R2), où R1 désigne la représentation qui sera utilisée pour la situation de départ et R2 la représentation de la situation qui correspond à la situation de départ.

A chacun des couples de représentation qu'il veut mettre en présence, l'enseignant doit associer un nombre d'exercices (compris entre 0 et 25). Il ne pourra pas choisir les représentations des deux situations qui ne correspondent pas à la situation de départ. Mais il peut être certain que celles-ci appartiendront aux lots des représentations qu'il aura choisi d'utiliser.

3.2.3. Exercice 3.

Les sous-types de l'exercice sont constitués par le nombre de correspondances qui existent entre les situations:

- aucune correspondance;
- les situations se correspondent deux à deux;
- deux situations se correspondent, les deux autres pas;
- trois situations se correspondent;
- les quatre situations se correspondent.

A chacun des sous-types qu'il veut mettre en oeuvre, l'enseignant doit associer un nombre d'exercices compris entre 0 et 25.

3.3. Choix concernant les valeurs numériques.

Les valeurs numériques de A,B,C ont peu d'importance en elles-mêmes, mais elles ne doivent pas constituer un obstacle pour l'élève.

1. L'enseignant peut, d'une part, choisir la valeur maximale que la somme C pourra atteindre. Nous prévoyons que cette valeur ne puisse pas dépasser cent, ni être inférieure à dix.

2. D'autre part, il pourra éviter l'apparition de situations où le calcul de A + B nécessite un passage à la dizaine, tels que '14 + 9', ou '24 + 37'.

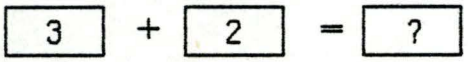
3.4. Choix concernant les situations problématiques.

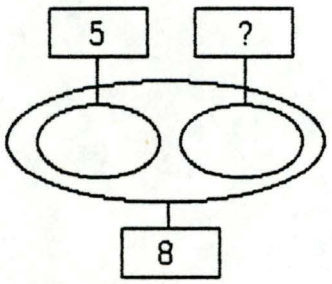
3.4.1. Choix pour les situations qui se correspondent.

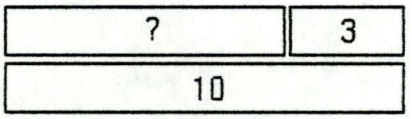
Dans les représentations telles que nous les avons dessinées, les localisations des parties A et B sont toujours bien définies. Lorsque deux situations S1 et S2 se correspondent, l'enseignant peut choisir que les localisations de A et de B soient parfois inversées dans le dessin des représentations de S1 et de S2. Ce choix se fait en désignant les représentations pour lesquelles on laisse la possibilité d'inverser les parties.

Exemples d'inversion.

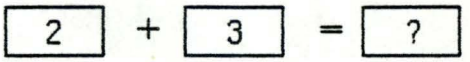
S1

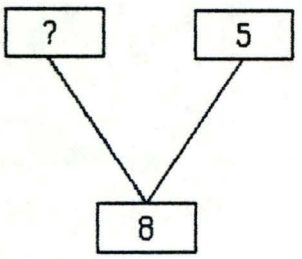


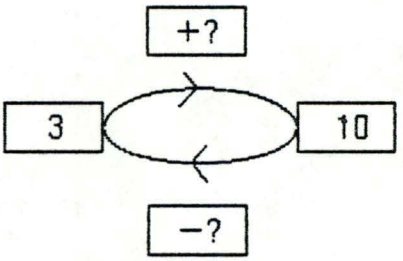




S2







3.4.2. Choix concernant les situations qui ne se correspondent pas.

Pour que deux situations ne se correspondent pas, il faut qu'il y ait des distorsions entre ces situations. L'enseignant pourra sélectionner les types distorsions qu'il veut introduire dans les exercices, parmi les listes de distorsions présentées dans les points suivants.

3.4.2.1. Les situations qui se correspondent sont de la forme A + B = ?.

1. Utiliser les mêmes nombres, et créer une nouvelle situation problématique:

   A + ? = B (si B>A)
   ? + B = A (si B<A)

2. Changer une valeur connue:
   - A ou B est remplacé par la valeur de l'inconnue C:

     C + B = ? A + C = ?
     ? + B = C A + ? = C

   - A ou B est remplacé par une valeur quelconque X:

     X + B = ? A + X = ?
     ? + B = X A + ? = X
     (X>B) (X>A)
     ? + X = B X + ? = A
     (X<B) (X<A)

3. Changer 2 valeurs connues:
   - A et B remplacés par C et X.(X ≠ A, B, C):

     X + C = ?
     ? + C = X X + ? = C
     (X>C) (X<C)

   - A et B remplacés par X et Y.(X, Y ≠ A, B, C et X > Y):

     X + Y = ?
     ? + Y = X

3.4.2.2. Les situations qui se correspondent sont de la forme A + ? = C.

1. Utiliser les mêmes nombres et créer une nouvelle situation:

   A + C = ?

2. Une valeur connue change :
   - A ou C est remplacé par B.

   C + B = ? A + B = ?
   ? + B = C A + ? = B ? + B = A
              (A<B)   (A>B)

   - A ou C est remplacé par X:(X ≠ A ≠ C ≠ B)

   X + C = ? A + X = ?
   ? + C = X A + ? = X
   (X>C)   (X>A)
   ? + X = C X + ? = A
   (X<C)   (X<A)

3. Deux valeurs connues changent:
   - A et C remplacés par B et X.(X ≠ A ≠ C ≠ B)

   X + B = ?
   ? + B = X X + ? = B
   (X>B)   (X<B)

   - A et B remplacés par X et Y.(X > Y).

   X + Y = ? ? + Y = X

Le raisonnement pour ´? + B = C´ est le même que pour ´A + ? = C´ où A est remplacé par B et B par A.

L´enseignant devra au moins choisir deux distorsions dans chacune des listes pour chaque exercice.

L´ordre des termes n´a aucune importance dans chacune des expressions.

4. Description d'une session d'exercices.

Une session d'exercices est définie par une préparation associée à un type d'exercice. Chaque session présente donc des exercices du type exercice 1, exercice 2, exercice 3 décrits au point 2.1. Le nombre d'exercices présenté est fixé dans la préparation. Il sera toujours possible d'interrompre prématurément une session en appuyant sur la touche 'escape'. Lorsque la session est terminée (prématurément ou non), on affiche à l'écran le nombre d'exercices qui ont été proposés à l'élève et le nombre d'exercices auxquels il a correctement répondu.

Les exercices composant la session sont générés de façon aléatoire, mais dans les limites des choix imposés par la préparation. Une préparation donne lieu à des sessions d'exercices fort semblables, mais toujours différents.

L'enchaînement des exercices est le suivant :

Un exercice est proposé à l'élève, une consigne lui est adressée en bas de l'écran l'invitant à résoudre l'exercice.

- Si la réponse de l'élève est correcte, un message de félicitation apparaît au bas de l'écran et l'on passe à l'exercice suivant.

- Si la réponse de l'élève n'est pas correcte, on lui montre la bonne solution et un message l'amène à réfléchir cette solution.En cas de problème, l'élève peut faire appel à son professeur. Le même exercice est ensuite présenté à l'élève, seules changent les valeurs numériques et la position des situations dans les fenêtres. L'élève est invité à répondre: s'il réussit, on enchaîne avec un nouvel exercice; sinon, on lui montre la bonne solution et une consigne lui demande de faire appel à l'enseignant.

5. Description du matériel.

L'école primaire de Floreffe ne dispose pas de matériel informatique. Nous avons utilisé le matériel disponible à l'Institut d'Informatique : l'Apple 2. Nous avons choisi de développer le didacticiel dans un langage de haut niveau, le Pascal pour qu'il soit accessible aux enseignants ayant une formation informatique. Sur l'Apple 2, il s'agit du PASCAL UCSD développé par l'Université de Californie de San Diego avec des variantes mineures par rapport au Pascal du professeur N. Wirth.

Caractéristiques du poste de travail.

- système	APPLE 2
- mémoire	48 K
- langage	PASCAL UCSO
- cartes	carte langage
- clavier	QWERTY
- écran	écran vidéo noir et blanc
- périphériques	2 unités disquettes 5 1/4

6. La maquette.

Une maquette est une réalisation à échelle réduite d'un projet. C'est une manière directe pour l'utilisateur de prendre conscience de la solution proposée, et de s'assurer qu'elle correspond bien à son attente.Une maquette permet de valider et d'améliorer les spécifications d'un projet.

Il nous a paru intéressant de prendre à notre compte l'idée du maquettage pour concrétiser les exercices imaginés sur papier: s'assurer de leur lisibilité , expérimenter le mode de dialogue avec l'utilisateur et mettre au point l'enchaînement des exercices. Cette maquette consistait en une simulation d'une série d'exercices. Elle comprenait donc l'affichage des diverses représentations utilisées, s'occupait de la gestion des réponses et offrait une version simplifiée de scénario de session.

Cette maquette n'a donné lieu qu'à des modifications de détails concernant le dessin de certaines représentations. Elle a surtout eu le mérite de donner une vue dynamique des exercices et donc a facilité la discussion pour établir le scénario d'enchaînement des exercices.

Cette maquette nous a donné l'occasion d'un premier contact avec l' Apple 2. Elle avait été développée pour tourner avec un écran couleur. Malgré les problèmes qu'elle a permis de déceler à ce niveau, une grande partie du code de la maquette a été récupérée, le blanc et le noir pouvant être considérés comme deux couleurs particulières.

Chapitre 4: ANALYSE ORGANIQUE

1. Introduction.

En guise d'introduction, nous présentons brièvement la méthode d'analyse enseignée par A. Van Lamsweerde et que nous avons utilisée pour cette étape. Le lecteur désireux d'en apprendre davantage pourra se référer à l'ouvrage [MYE;75]. Nous suggérons au lecteur ayant déjà connaissance de cette méthode de reprendre sa lecture au point suivant.

L'analyse organique est l'ensemble des étapes qui suivent l'analyse fonctionnelle et qui précèdent la mise en oeuvre du système. L'objectif de l'analyse organique est de donner une structure au logiciel. Une structure est une vue partielle du système définie par ses composants et les relations entre composants. L'étape d'analyse organique est subdivisée en plusieurs sous-étapes :

- la décomposition du système en modules.
- la hiérarchisation.

1.1. Décomposition du système en modules.

1.1.1. Objectifs de l'étape.

- faciliter la conception par une factorisation du travail.
- faciliter la modifiabilité, la maintenance.
- faciliter la validation.

1.1.2. La démarche.

Selon les objectifs, un module sera considéré comme une unité de conception, de modifiabilité, de validation.

Les modules seront les plus indépendants possibles :

- au point de vue travail, on ne doit rien connaître de ce qui est interne aux autres modules.
- au point de vue modifiabilité, un changement ne concerne qu'un module.

Les interfaces entre les modules sont constituées uniquement des hypothèses que celui qui écrit un module fait au sujet des autres modules. On essaiera d'avoir des interfaces minimaux qui ne changent pas lorsque le module est modifié.

La décomposition se fera selon 2 critères :

- l'"information hiding" ou "secret" et
- la quantité de travail.

On procède dès lors de la manière suivante :

Après étude du système, on identifie les décisions de conception qui sont susceptibles de changer. Ce seront "les secrets". Chaque secret sera enfermé dans un module. On dira que le module cache de l'information, car l'interface ne montre rien de ce secret.

Ensuite, on regarde si les modules ainsi construits sont assez petits comme unités de travail.

Les fonctions qui n'auront pas encore été prises seront mises dans des modules en tenant compte de l'indépendance, de la modifiabilité et de la grosseur du module.

On peut regrouper des fonctions communes dans un même module.

1.2. Hiérarchisation UTILISE.

1.2.1. Objectifs de l'étape.

Obtenir des systèmes souples c'est-à-dire

- contractables : on voudrait extraire des morceaux du système pour les réutiliser et
- extensibles : on voudrait pouvoir rajouter facilement de nouveaux services.

1.2.2. Démarche.

Nous allons regarder pour chaque module, les autres modules qu'il utilise. Soit un module A avec sa spécification S et un module B, A utilise B si A ne peut satisfaire S sans la présence correcte de B.

La définition de UTILISE permet de hiérarchiser les modules. Nous allons procéder à une décomposition en niveaux, les cycles n'étant pas permis :

- niveau 1 : modules n'utilisant pas d'autres modules.
- niveau i : modules utilisant seulement les modules de niveau < i.

La hiérarchie utilise permet :

- une conception factorisée : si A utilise B, lors de la conception de A, on fait abstraction de la manière dont B est réalisé; de même, lors de la conception de B, on ignore A.
- une validation factorisée en commencant par le niveau 1 et en remontant (bottom-up).

2. Analyse organique de la fonction préparation.

2.1. Introduction.

Pour l'analyse de la fonction de préparation, nous avons repris la démarche générale de hiérarchisation d'un système d'information à l'aide de la relation UTILISE. Cette hiérarchisation aboutit à une découpe en six niveaux.

2.1.1. Le niveau 6 : les modules fonctionnels.

Les modules fonctionnels se dérivent directement ou indirectement de l'analyse fonctionnelle. Les composés dérivés indirectement s'obtiennent soit par une décomposition plus fine, soit par composition. La décomposition sépare au sein d'une même fonction de l'analyse fonctionnelle des tâches dont les réalisations dynamiques doivent êtres distinguées, ou qui réalisent des types d'opérations différents. La composition consiste à regrouper en un seul module des fragments dispersés dans différentes fonctions de l'analyse fonctionnelle alors qu'ils sont liés à la production d'un même résultat ou à la réalisation d'un même concept.

2.1.2. Le niveau 5 : les noyaux fonctionnels.

Ils constituent l'ensemble des composants de base utilisés par les modules fonctionnels.

2.1.3. Le niveau 4 : les modules d'entrées sorties.

Ces modules comprennent la saisie de données, le contrôle de validité, le recouvrement d'erreurs, les accès à la base de données, etc ...

2.1.4. Le niveau 3 : les modules de contrôle.

Ils assurent le comportement dynamique correct du système. Ce niveau n'apparaîtra pas dans l'analyse organique de la fonction préparation.

2.1.5. Le niveau 2 : les modules outils.

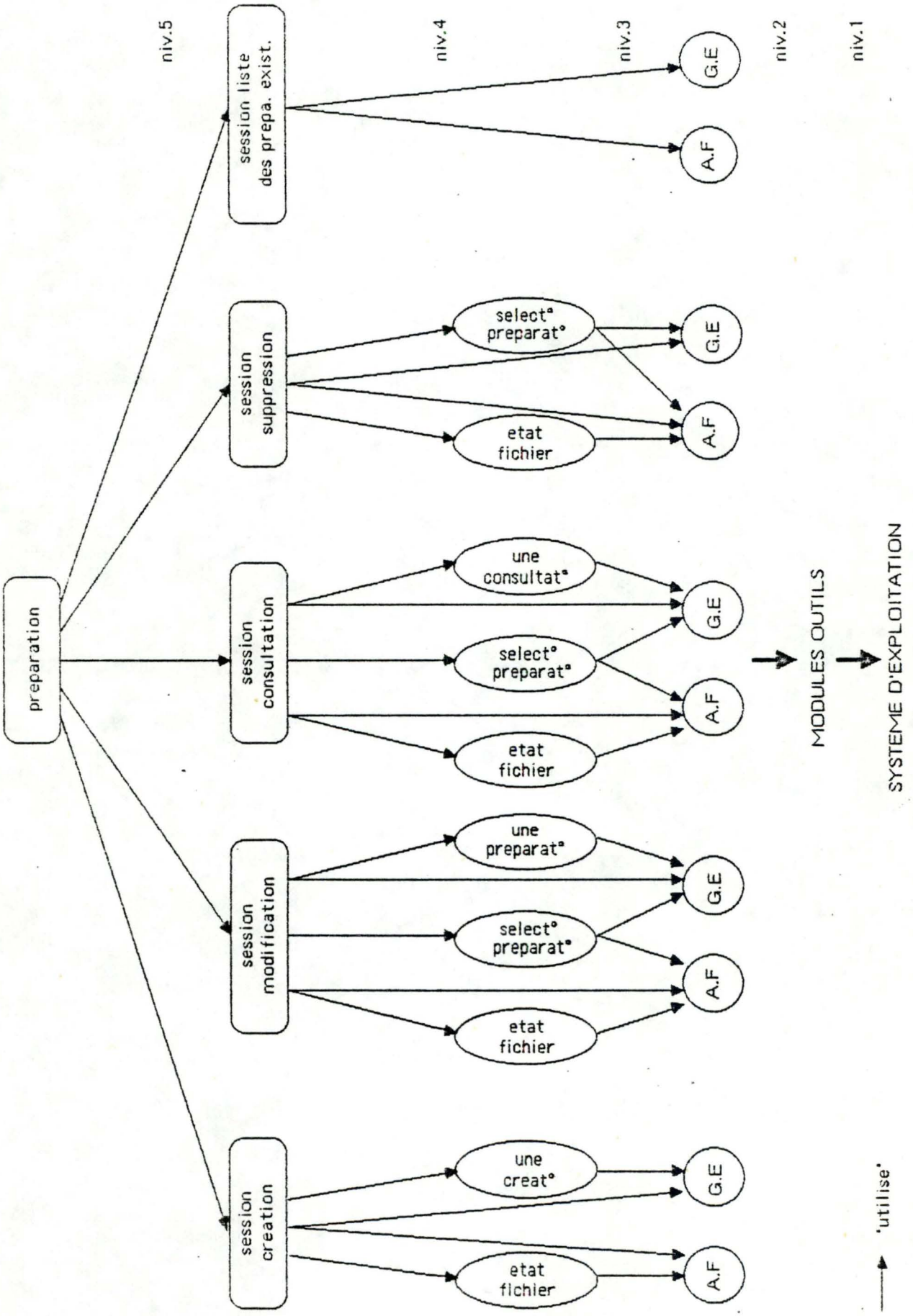
C'est l'ensemble des outils existants et disponibles. Ils sont visibles à l'utilisateur par des interfaces bien définis :

- nom de procédure et liste de paramètres.
- description des effets d'une utilisation.

2.1.6. Le niveau 1 : les modules du système d'exploitation.

Ce niveau peut lui-même faire l'objet d'une hiérarchisation à plusieurs niveaux.

2.2. Découpe en niveaux de la fonction préparation.



2.2.1. Le niveau 5.

Les modules fonctionnels obtenus par décomposition de la fonction de préparation sont :

- session création.
- session modification.
- session consultation.
- session suppression.
- session liste des préparations existantes.

Une session peut comporter plusieurs tâches de types différents :

- choisir un type d'exercice.
- choisir une préparation pour un type d'exercice.
- vérifier la place sur le fichier.
- effectuer des accès fichiers.
- effectuer la création, la modification, la consultation ou la suppression d'une préparation particulière.
- envoyer des messages à l'utilisateur.

Le secret de chacun des modules de niveau 5 réside dans le choix des tâches à réaliser et dans le choix de la séquence d'exécution de ces tâches. On insistera sur le fait que les modules de niveau 5 ignorent la manière dont ces tâches sont réalisées.

2.2.1. Le niveau 5.

Les modules fonctionnels obtenus par décomposition de la fonction de préparation sont :

- session création.
- session modification.
- session consultation.
- session suppression.
- session liste des preparations existantes.

Une session peut comporter plusieurs tâches de types différents :

- choisir un type d'exercice.
- choisir une préparation pour un type d'exercice.
- vérifier la place sur le fichier.
- effectuer des accès fichiers.
- effectuer la création, la modification, la consultation ou la suppression d'une préparation particulière.
- envoyer des messages à l'utilisateur.

Le secret de chacun des modules de niveau 5 réside dans le choix des tâches à réaliser et dans le choix de la séquence d'exécution de ces tâches. On insistera sur le fait que les modules de niveau 5 ignorent la manière dont ces tâches sont réalisées.

2.2.2. Le niveau 4.

Les noyaux fonctionnels sont :

- état fichier.
- sélection préparation.
- une création.
- une modification.
- une consultation.

Remarque sur les modules : 'une consultation', 'une création', 'une modification'.

Chacun de ces modules connaissent et cachent :

1. La sémantique liée à une préparation c'est-à-dire :
   - le nom et les valeurs des différents choix possibles pour une préparation.
   - les contraintes d'intégrités liées à ces choix.
2. La structure des grilles d'écrans et le scénario d'enchaînement.

2.2.3. Le niveau 3.

Le niveau 3 est le niveau des modules d'entrées sorties.

- accès fichier s'occupe des E/S sur le fichier des préparations. vv.S1
- gestion d'écran s'occupe des E/S avec l'utilisateur. Il connait et cache la manière dont est réalisé l'affichage des grilles d'écrans, la gestion du curseur, la réception et le contrôle des informations introduites au clavier.

2.2.4. Le niveau 2.

Ce niveau contient les utilitaires standards disponibles.

2.2.5. Le niveau 1.

Ce niveau représente le système d'exploitation.

2.3. Description des modules de la fonction préparation.

2.3.1. Glossaire.

numéro de fichier :

1 : fichier de préparation de l'exercice 1.
2 : fichier de préparation de l'exercice 2.
3 : fichier de préparation de l'exercice 3.

préparation :

- numéro de préparation.
- nom de préparation.
- représentation utilisée.
- orientations.
- dessins variables.
- nombre d'exercices.
- valeur maximale de la somme.
- report à la dizaine.
- inversions.
- distorsions liste 1.
- distorsions liste 2.

code nettoyage :

1 : lignes 0 à 23 (tout l'écran).
2 : lignes 0 à 17 (la fenêtre de présentation).
3 : lignes 19 à 23 (la fenêtre de dialogue).
4 : lignes 2 à 17.
5 : lignes 19 à 22.

code message : MS1 -> MS8.

code présentation : P1 -> P38.

code dialogue : D1 -> D18.

code menu : M1 -> M9.

numéro de zone :

dans chaque fenêtre de présentation,
les zones sont numérotées à partir de 1.

valeur à afficher :

'OUI' / 'NON' / nom de préparation / val max / nombre d'exercices .

sélection : entier.

valeur saisie :

'O' / 'N' / nom de préparation / nombre / chiffre.

code fin :

1 : return.
2 : escape.

numéro sélectionné : entier.

nombre préparations : 1 .. 10.

mode :

1 : affichage.
2 : modification.
3 : suppression.
4 : exécution.

type d'exercice :

1 : exercice 1.
2 : exercice 2.
3 : exercice 3.

2.3.2. Les modules outils.

Par modules outils nous entendons les différentes procédures et fonctions disponibles dans la librairie du système [APP;80].

Nous utilisons la fonction KEYPRESS contenue dans la librairie APPLESTUFF.

2.3.3. Les modules d'entrées sorties.

2.3.3.1. accès fichier.

| ouvre |

Entrée : - numéro de fichier.

fonction : - ouvrir le fichier des préparations correspondant au numéro de fichier.

| ferme |

Entrée : - numéro de fichier.

Fonction : - fermer le fichier des préparations correspondant au numéro de fichier.

| écriture |

Entrée : - numéro de fichier.
- une préparation.

fonction : - enregistrer la préparation dans le fichier correspondant au numéro de fichier.

| lecture |

Entrée : - numéro de fichier.
- numéro de préparation.

Sortie : - une préparation.

Fonction : - lire sur le fichier des préparations correspondant au numéro de fichier, la préparation correspondant au numéro de préparation.

2.3.3.2. Gestion d'écran.

Pour dialoguer avec l'utilisateur, nous utilisons:

- les grilles d'écrans. (voir annexes)
- la gestion des réponses utilisateurs.

1. Les grilles d'écrans sont de plusieurs types :

- les messages.
- les menus.
- les fenêtres.

Les fenêtres sont uniquement utilisées pour présenter les informations contenues dans une préparation . L'écran est divisé en deux parties par une ligne pointillée horizontale. C'est dans la partie inférieure que s'effectuera la majorité du dialogue avec l'utilisateur. On appelera cette fenêtre : 'fenêtre de dialogue'. La fenêtre supérieure est essentiellement dédicacée à la présentation des informations, c'est la 'fenêtre de présentation'.

Les fenêtres de dialogue permettent d'adresser des consignes à l'utilisateur et de lui poser des questions.

Toutes les fenêtres de présentation sont constituées de deux parties :

- une partie fixe, qui constitue l'emballage nécessaire à la présentation agréable de l'information (titres, commentaires, etc ...).

- une partie variable, qui constitue l'information proprement dite.

init fenêtre

fonction : - séparer l'écran en deux fenêtres par une ligne pointillée.

net écran

Entrée : - code nettoyage.

Fonction : - nettoyer l'écran en fonction du code de nettoyage.

aff message

Entrée : - code message.

Fonction : - afficher le message correspondant au code message.

aff menu

Entrée : - code menu.

Fonction : - afficher le menu correspondant au code menu.

aff présentation

entrée : - code présentation.

fonction : - afficher la fenêtre de présentation correspondant au code présentation.

|aff dialogue|

entrée : - code dialogue.

fonction : - afficher la fenêtre de dialogue correspondant au code dialogue.

|aff valeur|

entrée : - valeur à afficher.
- code présentation.
- numéro de zone.

fonction : - afficher la valeur à afficher dans la zone (correspondant au numéro de zone) de la fenêtre de présentation (correspondant au code de présentation).

2. La gestion des réponses de l'utilisateur.

La saisie de données comporte l'affichage des caractères (écho), la gestion du curseur, l'affichage des zones de réponses (pointillés), et le contrôle de validité des valeurs rentrées. Ce contrôle est effectué caractère par caractère. A un instant donné, seul un certain nombre de touches sont accessibles à l'utilisateur, la pression des autres touches restera sans effets.

saisie menu

entrée : - code menu.

sortie : - sélection.

fonction : - saisir la sélection effectuée par l'utilisateur sur le menu correspondant au code menu.

saisie message

fonction : - effectuer la saisie de 'return'.

saisie fenêtre

entrée : - code présentation. (P16 à P38)
- code dialogue. (D1 à D15)

sortie : - code fin.
- valeur saisie.

fonction : - effectuer la saisie de la valeur en fonction du code présentation et du code dialogue.
Renvoyer le code fin.

| saisie selection |

entrée : - code présentation. (P1 à P15)
- code dialogue. (D16 à D18)
- nombre préparation.

sortie : - code fin.
- numéro sélectionné.

fonction : - effectuer la saisie du numéro sélectionné par l'utilisateur et renvoyer le code fin.

2.3.4. Les noyaux fonctionnels.

|état fichier|

Utilise : - accès BD : ouvre
                       lecture
                       ferme.

Entrée : - numéro de fichier.

Sortie : - nombre de préparations dans le fichier.

Fonction : - renvoyer le nombre d'éléments du fichier
             correspondant au numéro de fichier.

|sélection préparation|

Utilise : - accès BD : ouvre
                       lecture
                       ferme.
          - gestion d'écran : init fenêtre
                              aff dialogue
                              aff valeur
                              aff présentation
                              saisie sélection.

Entrée : - numéro de fichier.
         - mode.
         - nombre de préparations du fichier.

sortie : - numéro de préparation.
         - OK : booléen.

Fonction : - renvoyer le numéro de préparation sélectionné
             par l'utilisateur. Si une sélection a été
             effectuée, OK = VRAI, sinon OK = FAUX.

une création

Utilise : - Gestion d'écran : init fenêtre
net écran
aff présentation
aff dialogue
aff valeur
saisie fenêtre.

Entrée : - type d'exercice.

Sortie : - une préparation.
- FIN : booléen.

Fonction : - collecter les informations relatives à une préparation d'exercice correspondant au type d'exercice. Si la collecte n'a pas été interrompue, FIN = FAUX, sinon FIN = TRUE.

une modification

Utilise : - Gestion d'écran : init fenêtre
net écran
aff présentation
aff dialogue
aff valeur
saisie fenêtre.

Entrée : - type d'exercice.
- une préparation à modifier.

Sortie : - une préparation modifiée.

Fonction : - pour toutes les valeurs qui composent la préparation laisser à l'utilisateur la possibilité de les modifier ou d'interrompre la modification.

|une consultation|

Utilise : - Gestion d'écran : init fenêtre
net écran,
aff dialogue
aff présentation
saisie fenêtre.

Entrée : - type d'exercice.
- une préparation à consulter.

Fonction : - laisser la possibilité de consulter toutes les informations relatives à la préparation et laisser la possibilité d'interrompre la consultation.

2.3.5. Les modules fonctionnels.

|session création|

Utilise : - accès fichier : ouvrir
          écriture
          ferme.
        - gestion d'écran : aff menu
          aff message
          saisie message
          saisie menu
          net écran.
        - état fichier.
        - une création.

Fonction : - réaliser toutes les opérations
           nécessaires à la création d'une préparation.

|session modification|

Utilise : - accès fichier : ouvrir
          lecture
          écriture
          ferme.
        - gestion d'écran : aff menu
          aff message
          saisie menu
          saisie message
          net écran.
        - état fichier.
        - sélection préparation.
        - une consultation.

Fonction : - réaliser toutes les opérations nécessaires
           à la modification d'une préparation.

|session consultation|

Utilise : - accès fichier : ouvrir
lecture
ferme.
- gestion d'écran : aff menu
saisie menu
net écran.
- état fichier.
- sélection préparation.
- une consultation.

Fonction : - réaliser toutes les opérations nécessaires à la consultation d'une préparation.

|session suppression|

Utilise : - accès fichier : ouvrir
lecture
écriture
ferme.
- gestion d'écran : net écran
aff menu
saisie menu
saisie message
aff message.
- état fichier.
- sélection préparation.
- une consultation.

Fonction : - réaliser toutes les opérations nécessaires à la suppression d'une préparation.

session liste des préparations existantes

Utilise : - accès fichier : ouvrir
lecture
ferme.
- gestion d'écran : net écran
init fenêtre
aff fenêtre présentation
aff valeur
saisie menu
saisie fenêtre
aff menu.

Fonction : - réaliser toutes les opérations nécessaires pour lister les préparations existantes pour un type d'exercice.

3. Analyse organique de la fonction exécution.

3.1. Introduction.

La fonction exécution des exercices comporte une partie graphique importante qui concerne essentiellement le dessin des différentes représentations. Pour décharger la fonction exécution des problèmes graphiques, nous les avons localisé dans un interface graphique. Cet interface est chargé d'offrir des primitives de haut niveau aux modules utilisateurs. Il peut faire lui même l'objet d'une hiérarchisation en module. Les différents modules constituant l'interface ont été conçus de manière à pouvoir être utilisés dans un autre contexte, ou pour la réalisation d'autres types d'exercices.

Avant d'analyser la fonction préparation, nous présenterons l'analyse de l'interface graphique.

3.2. L'interface graphique.

3.2.1. Découpe en niveaux.

3.2.1.1. Le niveau 1.

Constitué par le système d'exploitation.

3.2.1.2. Le niveau 2.

Le niveau contient les utilitaires standards disponibles.

Nous utilisons dans la librairie TURTLEGRAPHICS :

- INITTURTLE;
- GRAFMODE;
- TEXTMODE;
- VIEWPART;
- PENCOLOR;
- FILLSCREEN;
- TURNTO;
- TURN;
- MOVE;
- MOVETO;
- DRAWBLOCK;
- WCHAR;
- WSTRING.

Dans la librairie TRANSCEND:

- SIN;
- COS.

3.2.1.3. Le niveau 3.

1. Les modules de dessin d'objets simples :

   - deslig : dessin d'une droite.
   - desarc : dessin d'un arc.
   - desrec : dessin d'un rectangle.
   - colrec : coloriage d'un rectangle.
   - ellipse : dessin d'une ellipse.

   Chacun de ces modules cache la manière dont il réalise le dessin.

2. Les modules permettant d'afficher un caractère ou une phrase sur l'écran graphique :

   - affcar.
   - affphr.

3. Les modules pour afficher et effacer un petit bonhomme :

   - affbon.
   - effbon.

3.2.1.4. Le niveau 4.

1. Les modules de gestion des fenêtres :

   - créer fenêtre " utilise " colrec.
   - créer fenêtrel " utilise " colrec.
   - eff fenêtre " utilise " colrec.
   - eff fenêtrel " utilise " colrec.

2. Les modules de dessin de situations dans un mode de représentation :

   - diagramme " utilise " deslig, desrec, ellipse, affphr.
   - arbre " utilise " deslig, desrec, affphr.
   - graphe " utilise " desrec, desarc, affcar, affphr.
   - lineaire " utilise " desrec, affphr.

- artpos " utilise " desrec
  affcar
  affphr.

- artneg " utilise " desrec
  affcar
  affphr.

3. Un module d'animation du bonhomme :

   - animbom " utilise " affbon
     effbon.

4. Un module permettant l'envoi de consignes et de commentaires :

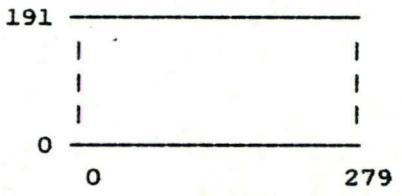
   - commentaire " utilise " affphr.

3.2.2. Description des modules.

3.2.2.1. Glossaire.

La description des écrans et des dessins des représentations se trouvent en annexe.

écran : ensemble des points sur la vidéo allant de 0 à 279 en largeur de 0 à 191 en hauteur.



axe des x : axe des coordonnées en x des points de l'écran allant de 0 à 279.

axe des y : axe des coordonnées en y des points de l'écran allant de 0 à 191.

couleur :
0 pas de couleur.
1 blanc.
2 noir.
3 noir1.
4 vert.
5 violet.
6 blanc1.
7 orange.
8 bleu.
9 blanc2.

orientation : N nord.
S sud.
E est.
O ouest.

phrase : string.

numèro fenètre : 1 à 5.

numèro ligne 1 à 4.

message : string.

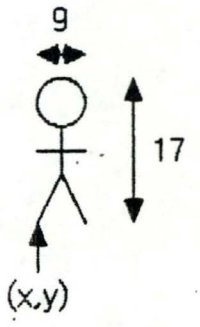
etiquette: string.

code représentation diagramme : 1 à 3.

code représentation graphe : 1 à 2.

3.2.2.2. Le niveau 3.

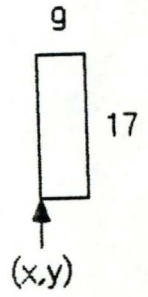
affbon



entrée : X coordonnée sur l'axe des x.
Y coordonnée sur l'axe des y.

fonction : afficher à l'écran en (X,Y) un bonhomme d'une hauteur de 17 points d'une largeur de 9 points.

effbon



entrée : X coordonnée sur l'axe des x
Y coordonnée sur l'axe des y

fonction : effacer à l'écran en (X,Y) un rectangle d'une hauteur de 17 points d'une largeur de 9 points.

deslig

entrée : C couleur.
X1 coordonnée sur l'axe des x.
X2 coordonnée sur l'axe des x.
Y1 coordonnée sur l'axe des y.
Y2 coordonnée sur l'axe des y.

fonction : dessiner à l'écran dans la couleur C un segment de droite d'origine (X1,Y1) d'extrémité (X2,Y2).

affcar

entrée : CH caractère.
X coordonnée sur l'axe des x.
Y coordonnée sur l'axe des y.

fonction : afficher à l'écran CH en (X,Y).

affphr

entrée : X coordonnée sur l'axe des x.
Y coordonnée sur l'axe des y.
PHR phrase.

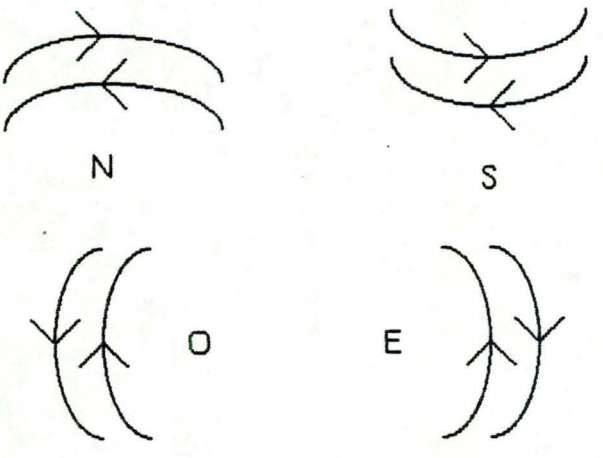
fonction : afficher la phrase PHR à l'écran en (X,Y).

desarc

entrée : C couleur.
X1 coordonnée sur l'axe des x.
X2 coordonnée sur l'axe des x.
Y1 coordonnée sur l;axe des y.
Y2 coordonnée sur l'axe des y.
RI orientation.

si RI = N ou S alors Y1=Y2
si Ri = E ou O alors X1=X2

fonction : dessiner à l'écran un arc flèché dans la couleur C , d'origine (X1,Y1), d'extrémité (X2,Y2), dans l'orientation RI.

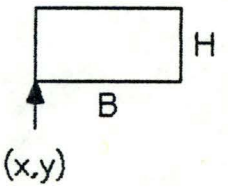


colrec

entrée : C couleur.
X coordonnée sur l'axe des x.
Y coordonnée sur l'axe des y.
B base du rectangle.
H hauteur du rectangle.

fonction : colorier à l'écran dans la couleur C un rectangle de base B, de hauteur H, en (X,Y).

desrec



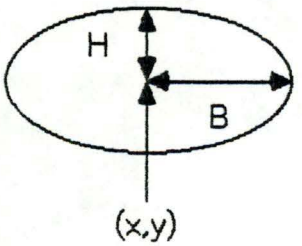
entrée : C couleur.
X coordonnée sur l'axe des x.
Y coordonnée sur l'axe des y.
B base du rectangle.
H hauteur du rectangle.

fonction : dessiner à l'écran dans la couleur C un rectangle de base B, de hauteur H, en (X,Y).

ellipse

entrée : C couleur.
X coordonnée sur l'axe des x.
Y coordonnée sur l'axe des y.
B base de l'ellipse.
H hauteur de l'ellipse.

fonction : dessiner à l'écran une ellipse de couleur C, dont le centre est situé en (X,Y) de base B et de hauteur H.



3.2.2.3. Le niveau 4

Les différents shèmas auxquels il est fait référence ici sont repris en annexe.

| eff fenêtre |

entrée : NO numéro de fenêtre.

fonction : effacer la fenêtre de numéro NO.

| eff fenêtre1 |

fonction : effacer la fenêtre 1 du shèma 1.

| créer fenêtre |

fonction : diviser l'écran en cinq fenêtres selon le shèma 1.

| créer fenêtre1 |

fonction : diviser l'écran en cinq fenêtres selon le shèma 2.

| commentaire |

entrée : NOLIG numéro de ligne.
MSG message.

Fonction : afficher le message MSG à la ligne de numéro NOLIG

| animbon |

entrée : NOD numéro de fenêtre.
------- NOO numéro de fenêtre.

fonction : effacer le bonhomme dans la fenêtre d'origine NOO et
-------- l'afficher dans la fenêtre de destination NOD.

| diagramme |

entrée : NO numéro de fenêtre.
------- CRD code de représentation diagramme.
SENS orientation.
CA étiquette.
CB étiquette.
CC étiquette
COLA couleur.
COLB couleur.
COLC couleur.

fonction : dessiner à l'écran une représentation en diagramme
-------- dans la fenêtre NO, selon SENS et CRD. Les cases A,B et C contiennent respectivement les étiquettes CA,CB et CC. Les cases A, B et C sont respectivement de couleur COLA, COLB et COLC. Le batonnet et l'ellipse reliés à une case ont la même couleur que la case.

| arbre |

entrée : NO numéro de fenêtre.
------- SENS orientation.
CA étiquette.
CB étiquette.
CC étiquette.
COLA couleur.
COLB couleur.
COLC couleur.

fonction : dessiner à l'écran une représentation en arbre
-------- dans la fenêtre NO, selon SENS. Les cases A, B et C contiennent respectivement les étiquettes CA,CB et CC.
La case C est de couleur COLC.
La case A et son batonnet sont de couleur COLA.
La case B et son batonnet sont de couleur COLB.

graphe

entrée : NO numéro de fenêtre.
CRG code de représentation graphe.
SENS orientation.
CA étiquette.
CB étiquette.
CC étiquette.
COLA couleur.
COLB couleur.
COLC couleur.

fonction : dessiner à l'écran une représentation en graphe dans la fenêtre NO, selon CRG et SENS. Les cases A,B et C contiennent respectivement les étiquettes CA,CB et CC . Les cases A,B et C sont respectivement de couleur COLA, COLB et COLC. Les flèches restent blanches.

linéaire

entrée : NO numéro de fenêtre.
SENS orientation (N ou S).
CA étiquette.
CB étiquette.
CC étiquette.
COLA couleur.
COLB couleur.
COLC couleur.

fonction : dessiner à l'écran une représentation linéaire dans la fenêtre NO selon SENS. Les rectangles sont de longueurs proportionnelles aux valeurs contenues dans les étiquettes. Les rectangles A, B et C contiennent respectivement CA, CB et CC. Ils sont respectivement de couleur COLA, COLB et COLC.

| artpos |

entrée : NO numéro de fenêtre.
SENS orientation.
CA étiquette.
CB étiquette.
CC étiquette.
COLA couleur.
COLB couleur.
COLC couleur.

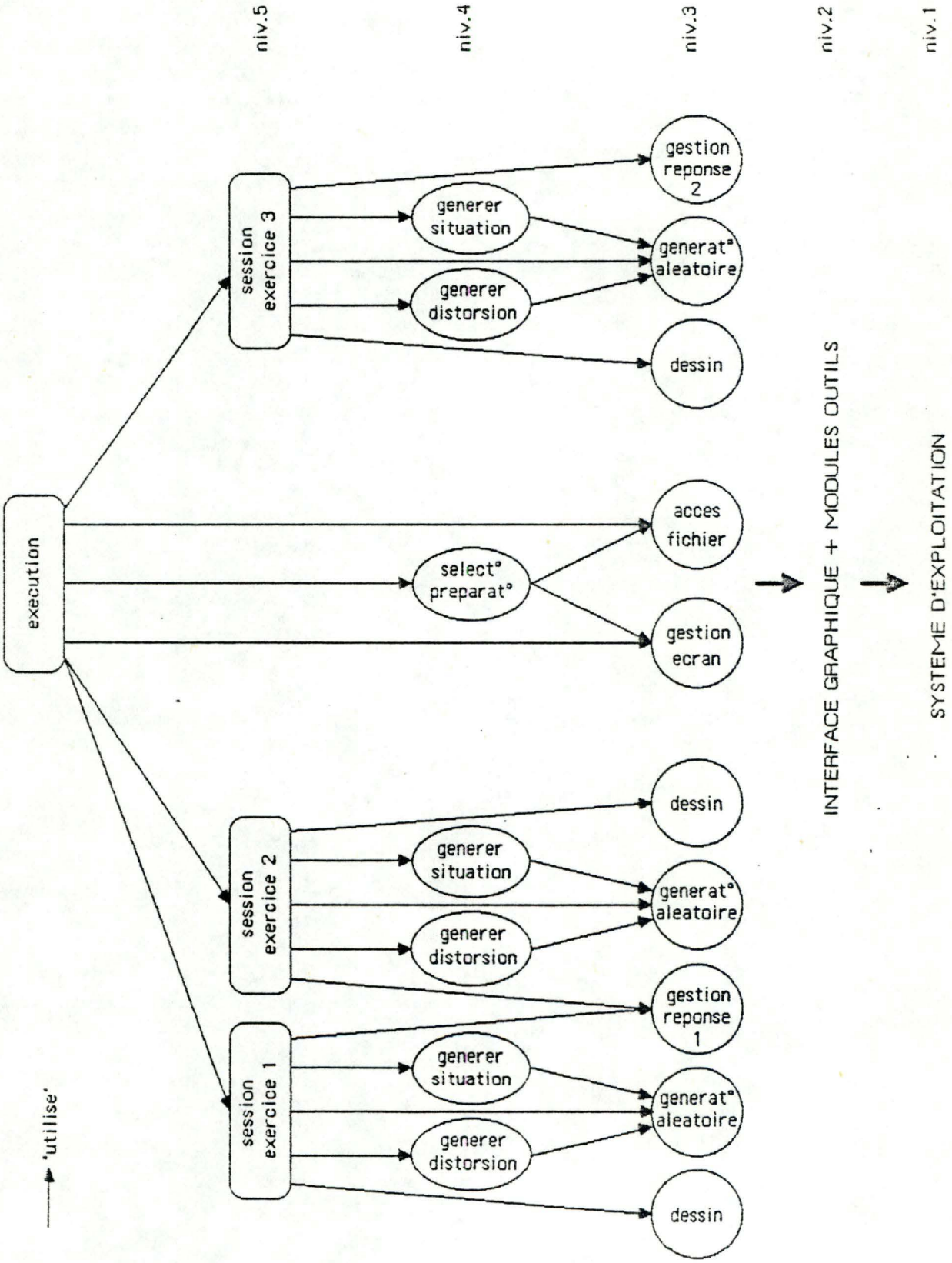
fonction : dessiner à l'écran une représentation arithmétique avec le signe '+' dans la fenêtre NO. les cases A, B et C contiennent les étiquettes CA, CB et CC. Elles sont respectivement de couleur COLA, COLB et COLC.

| artneg |

entrée : NO numéro de fenêtre.
SENS orientation.
CA étiquette.
CB étiquette.
CC étiquette.
COLA couleur.
COLB couleur.
COLC couleur.

fonction : dessiner à l'écran une représentation arithmétique avec le signe '-' dans la fenêtre NO. les cases A, B et C contiennent les étiquettes CA, CB et CC. Elles sont respectivement de couleur COLA, COLB et COLC.

3.3. Découpe en niveaux de la fonction exécution.



3.3.1. Le niveau 5.

Les modules fonctionnels obtenus par décomposition de la fonction exécution sont au nombre de trois :

- session exercice1;
- session exercice2;
- session exercice3.

Chacun de ces modules cache la manière dont un exercice est réalisé. Une session comporte :

- la génération des paramètres d'un exercice.
- le dessin de l'exercice à l'écran.
- la gestion de la réponse.

3.3.2. Le niveau 4.

Les noyaux fonctionnels sont :

- générer situation.
- générer distorsion.
- sélection préparation.

3.3.3. Le niveau 3.

Le niveau 3 comporte :

Les modules d'entrée sortie :

- gestion d'écran.
- accès fichier.
- gestion réponse 1.
- gestion réponse 2.
- dessin.

un module de génération aléatoire :

- génération aléatoire.

3.3.4. Le niveau 2.

Les modules outils :

- l'interface graphique.
- la librairie APPLESTUFF:
- la fonction KEYPRESS.
- la fonction RANDOM.
- la procédure RANDOMIZE.

3.3.5. Le niveau 1.

Ce niveau représente le système d'exploitation.

3.4. Description des modules de la fonction exécution.

3.4.1. Glossaire.

code de représentation :

D : diagramme.
G : graphe.
A : arbre.
L : linéaire.
+ : écriture arithmétique signe '+'.
- : écriture arithmétique signe '-'.

code association :

1 : association 2-1-1.
2 : association 2-2.
3 : association 3-1.
4 : association 4.
5 : aucune association.

valmax , valeur : 10 à 100.

report à la dizaine : booléen.

situation :

- A : valeur.
- B : valeur.
- C : valeur.
- INC : inconnue.

inconnue : 'A'/'B'/'C'.

code distorsion : 1 à 18.

table des numéros de fenêtres :

(numéro de fenêtre) x 4

table1 des exercices :

(CD , CA , NEX) x T1EX

CD : code de représentation de la situation de départ.
CA : code de représentation de la situation correspondante.
NEX : nombre exercice.
T1EX : entier. Représente la longueur de la table1.

table2 des exercices :

(CA , NEX) x T2EX

CA : code association.
NEX : nombre exercice.
T2EX : entier. Représente la longueur de la table2.

nombre exercice : 1 à 25.

table des inversions:

(CR , boulèen) x 6

CR : code représentation.

Le booléen associé à chaque représentation indique si l'inversion est autorisée ou non.

liste code diagramme :

Liste de longueur LCD qui contient les codes diagramme.
LCD : 0 à 12

code diagramme : D1 à D12.

liste code graphe :

Liste de codes graphe de longueur LCG.
LCG : 0 à 8.

code graphe : G1 à G8.

liste code arbre :

Liste de codes arbre de longueur LCA.
LCA : 0 à 4.

code arbre : A1 à A4.

liste code linéaire :

Liste de codes linéaire de longueur LCL.
LCL : 0 à 2.

code linéaire : l1 , l2.

liste code artpos :

Liste de codes artpos de longueur LC+.
LC+ : 0 à 2.

code artpos : EA+1 , EA+2.

liste code artneg :

Liste de code artneg de longueur LC-.
LC- : 0 à 2.

code artneg : EA-1 , EA-2.

liste1 des distorsions :

Liste de longueur LL1 qui contient les codes de distorsions (1 à 9) admis par rapport à une situation de départ
A + B = ?.

LL1 : 2 à 9.

liste2 des distorsions :

Liste de longueur LL2 qui contient les codes de distorsions (10 à 18) admis par rapport à une situation de départ
A + ? = C ou ? + B = C.

LL2 : 2 à 9.

code distorsion : 1 à 18.

code dessin : code diagramme/code graphe/
code arbre/code linéaire/
code artpos/code artneg.

dessin :

(CD , CASEA , CASEB , CASEC) x 4

CD : code dessin.
CASEA, CASEB, CASEC : étiquette.

intervalle :

- INF : entier.
- SUP : entier.

INF est inférieur ou égal à SUP.

nombre généré : entier.

3.4.2. Le niveau 3.

dessin

entrée : dessin .

fonction : dessiner dans les fenêtres 1,2,3 et 4, les quatre représentations selon le code de dessin CD, avec comme contenu des cases, 'caseA, caseB, caseC'.

gestion réponse 1

sortie : numéro de fenêtre. (2,3 ou 4).
fin : booléen.

fonction : renvoyer le numéro de fenêtre sélectionné par l'élève lors d'un exercice de type 1 ou 2. (cela implique aussi la gestion du petit bonhomme). Si l'utilisateur désire interrompre la session (touche ESC), fin = TRUE ,sinon fin = FALSE.

gestion réponse 2

sortie : code association.
liste de numéros de fenêtres .
fin : booléen.

fonction : renvoyer le type d'association et la liste des fenêtres mises en correspondance lors d'un exercice de type 3.(cela implique aussi la gestion du petit bonhomme). Si l'utilisateur veut interrompre la session (touche ESC) alors fin = TRUE, sinon fin = FALSE.

|génération aléatoire.|

entrée: intervalle.

sortie: nombre généré.

fonction: générer aléatoirement un nombre compris dans l'intervalle.

3.4.3. Le niveau 4.

|générer situation.|

entrée : val max.
report à la dizaine .

sortie : situation.

fonction : générer une situation, c'est- à- dire une valeur pour A,B,C et inconnu. De plus, si 'report à la dizaine' est FAUX, alors le calcul de A+B ne peut occasionner un report à la dizaine. Enfin, la valeur de C doit être comprise entre dix et "val max" et la somme de A et de B doit être égal à C.

|générer distorsion.|

entrée : situation
code distorsion.

sortie : situation.

fonction : étant donnée une situation de départ, générer une autre situation en fonction du code de distorsion.

3.4.4. Le niveau 5.

|sesssion exercice 1|

entrée : table1 des exercices , T1EX.
liste code diagramme , LCD.
liste code graphe , LCG.
liste code arbre , LCA.
liste code linéaire , LCL.
liste code artpos , LC+.
liste code artneg , LC−.
liste1 des distorsions , LL1.
liste2 des distorsions , LL2.
table des inversions.
val max.
report à la dizaine.

sortie : nombre d'exercices proposés.
nombre d'exercices corrects.

fonction : présenter une session d'exercices de type 1,
et renvoyer le nombre d'exercices proposés, et
le nombre d'exercices corrects.

|sesssion exercice 2|

entrée : table1 des exercices , T1EX.
liste code diagramme , LCD.
liste code graphe , LCG.
liste code arbre , LCA.
liste code linéaire , LCL.
liste code artpos , LC+.
liste code artneg , LC−.
liste1 des distorsions , LL1.
liste2 des distorsions , LL2.
table des inversions.
val max.
report à la dizaine.

sortie : nombre d'exercices proposés.
nombre d'exercices corrects.

fonction : présenter une session d'exercices de type 2,
et renvoyer le nombre d'exercices proposés, et
le nombre d'exercices corrects.

|sesssion exercice 3|

entrée : table2 des exercices , T2EX.
liste code diagramme , LCD.
liste code graphe , LCG.
liste code arbre , LCA.
liste code linéaire , LCL.
liste code artpos , LC+.
liste code artneg , LC-.
liste1 des distorsions , LL1.
liste2 des distorsions , LL2.
table des inversions.
val max.
report à la dizaine.

sortie : nombre d'exercices proposés.
nombre d'exercices corrects.

fonction : présenter une session d'exercices de type 3,
et renvoyer le nombre d'exercices proposés, et
le nombre d'exercices corrects.

IMPLEMENTATION 5.1

Chapitre 5: IMPLEMENTATION

1. Introduction.

L'étape d'implémentation a pour résultat un programme exécutable. Le texte de ce programme se trouve en annexe. L'objet de ce chapitre n'est pas d'expliquer le contenu du programme, mais plutôt de présenter la manière dont il a été structuré en tenant compte d'un côté de l'analyse organique, et de l'autre, des problèmes liés à l'implémentation de gros programmes Pascal sur Apple 2.

La mémoire limitée du micro-ordinateur pose en effet des problèmes lors de l'édition , de la compilation et de l'exécution d'un programme volumineux. Le Pascal sur Apple 2 offre différentes techniques pour pallier à ces inconvénients. Ces techniques sont décrites dans le premier point de ce chapitre. Nous suggérons au lecteur déjà averti de ces problèmes de passer au point suivant.

La suite du chapitre est consacrée à la présentation de la structure du programme. Nous aborderons successivement la partie préparation et la partie exécution pour expliquer ce qu'il advient des modules de l'analyse organique.

2. Création de gros programmes Pascal.

Lors de l'étape d'implémentation, nous avons rencontré des problèmes d'édition, de compilation et d'exécution dus au manque de place mémoire. Pour résoudre ces problèmes, nous avons eu recours à des techniques offertes par le Pascal Apple. Il nous semble intéressant de présenter ici ces techniques et ce, pour deux raisons: d'une part pour expliquer comment elles ont influencé la structure de nos programmes; d'autre part pour regrouper et résumer en un seul endroit des informations dispersées dans les manuels Pascal Apple.

2.1. Gestion de gros programmes sources.

Lors de la phase d'édition d'un programme, l'éditeur de texte et le programme sont présents en même temps en mémoire. Dès lors ,il n'y a souvent pas assez de place pour un programme en un seul fichier. Nous devons donc diviser le programme en un ensemble de petits fichiers que nous créons et modifions séparément. Le P-Systeme offre deux possibilités pour combiner plusieurs fichiers sources en un seul programme:

- Les fichiers INCLUDE.
- Les UNITs.

2.1.1. Le fichier INCLUDE.

Le fichier INCLUDE permet de combiner des parties de programme source qui ont été créées sous forme de fichiers séparés, en un seul programme, sans en changer la signification.

La directive INCLUDE ,(notée (*$I nom de fichier*)), indique au compilateur quel fichier doit être inclus et où doit se faire l'inclusion dans le programme principal. La notion de fichier INCLUDE constitue un outil intéressant mais possèdent un gros inconvénient: Si un seul des fichiers inclus est modifié, tous les fichiers doivent être recompilés pour produire une version modifiée du code objet.

2.1.2. La compilation séparée : les UNITs.

Une UNIT est un ensemble de procédures , de fonctions et de déclarations Pascal qui forment une section indépendante de programme. Un dispositif Pascal UCSD permet de réaliser la compilation séparée des UNITs. Ainsi, de nombreux programmes peuvent utiliser une même UNIT sans la recompiler.

Pour utiliser une UNIT un programme ,appellé programme hôte , doit contenir la directive USES suivie du nom de la UNIT.

Le résultat de la compilation d'une UNIT est une librairie. Cette librairie peut être utilisée directement ou être incorporée à une autre librairie comme la librairie SYSTEM.LIBRARY.

Il y a deux sortes de UNITs: les REGULAR UNITs et les INTRINSIC UNITs. Lorsqu'un programme utilise une REGULAR UNIT ,le code de la UNIT est insérè dans le code du programme hôte par l'éditeur de lien (LINKER). Lorsqu'un programme utilise une INTRINSIC UNIT, le code de la UNIT reste dans le fichier librairie et n'est pas intègrè au programme hôte. Le code de la UNIT est chargé en mémoire centrale au moment de l' exécution du programme hôte et l'édition de liens n'est donc pas nécessaire. Ce genre de UNIT est particulièrement intéressante lorsque beaucoup de programmes l'utilisent puisqu'elle permet de réduire la taille du fichier code du programme hôte.

2.2. Compilation de gros programmes sources.

Le problème majeur de la compilation de gros programmes, et en particulier des grosses UNITs, est la taille de la mémoire disponible pour recevoir le compilateur et la table des symboles.

Si durant la compilation l'espace mémoire n'est pas suffisant, le compilateur s'arrête ,indiquant une erreur de débordement de pile. Une première solution consiste alors à utiliser les options de recouvrement qui permettent de libérer de la place pour la table des symboles. Ce sont les options de compilation SWAP (noté (*$s+*)), et SWAP étendu (noté (*$s++*)) qui permettent de libérer 5000 et 6500 mots supplémentaires pour la table des symboles.

Si cela ne suffit pas, une autre technique consiste à restructurer le programme de façon à réduire la taille de la table des symboles. Concrètement cela signifie :

- ne déclarer des variables globales que lorsque c'est strictement nécessaire c'est-à-dire réduire la portée des identificateurs aux procédures qui les utilisent.

- réduire le nombre de paramètres associés aux procédures et aux fonctions.

- utiliser des types nommés.

- déclarer les procédures et les fonctions locales aux seules procédures et fonctions où elles sont utilisées.

2.3. L'exécution de gros programmes.

L'espace mémoire est également une ressource critique pour l'exécution de gros programmes. Pour remédier au problème nous pouvons agir au niveau des données et au niveau du code.

2.3.1. Réduire l'occupation mémoire des données.

l'UCSD offre deux possibilités :

- Augmenter l'usage des variables locales tout comme pour la compilation.
- Utiliser les structures de données compressées (PACKED FILE, PACKED ARRAY, PACKED RECORD)

2.3.2. La segmentation.

Le principe de la segmentation est de diviser le programme en parties , appelées segments. Le système peut amener un segment en mémoire indépendamment des autres. Pour gérer l' espace mémoire, il suffit de concevoir des segments indépendants les uns des autres et de taille importante. Lorsqu'un segment est nécessaire pour la bonne exécution du programme, il est chargé en mémoire. Le gain en place mémoire est compensé par une perte en temps d'exécution, occasionnée par les transferts des segments. Il est donc conseillé de réduire autant que possible le nombre de transferts.

Les segmentations peuvent être réalisées à l'aide des déclarations SEGMENT PROCEDUREs et des SEGMENT FUNCTIONs ,extensions du Pascal UCSD. Ces procédures et fonctions sont chargées en mémoire au moment de leur appel. Les REGULAR UNITs sont également des segments. Il est possible de contrôler le chargement en mémoire de ces segments grâce aux options de compilation NO LOAD et RESIDENT.

3. Structure du programme.

Le programme principal RECONNAITRE se compose du corps du programme et de deux parties indépendantes :

- La 'segment procedure' PREPARATION.
- La 'segment procedure' EXECUTION.

Les deux segments correspondent aux modules préparation et exécution de l'analyse organique. Le corps du programme principal est un simple aiguillage sur le mode préparation ou exécution, selon le choix de l'utilisateur.

3.1. Implémentation de la partie préparation.

La 'segment procedure' PREPARATION se compose :

- de trois segments indépendants :
  - la 'segment procedure' CREER.
  - la 'segment procedure' AFFICHER.
  - la 'segment procedure' MODIFIER.
- d'un ensemble de procédures :
  - SELMODE
  - SELEX
  - SELPREPA
  - MESSAGE
  - SELMODIF
- du corps principal de la procédure.

Nous reprenons une étude par niveau de l'identité physique des modules de l'analyse organique.

3.1.1. Niveau 2.

Les modules outils sont déjà implémentés et disponibles dans la librairie SYSTEM.LIBRARY.

3.1.2. Niveau 3.

Le niveau 3 comprend les modules de gestion d'écran et d'accès fichiers. Pour l'implémentation de ces modules, nous avons suivi deux démarches distinctes.

D'une part, il semblait intéressant de regrouper au sein de la librairie système des petites procédures fréquemment invoquées tant dans la partie préparation qu'exécution. Ces procédures deviennent ainsi très facilement accessibles à tous les niveaux du programme. Il s'agit de la UNIT FICHIER et de la UNIT ECRAN.

La UNIT FICHIER correspond aux modules d'accès fichiers. Elle contient les procédures

- OUVRE
- FERME
- LECTURE
- ECRITURE

La UNIT ECRAN correspond aux modules de gestion d'écran init fenêtre et net écran. Elle contient les procédures

- LIGNE
- EFFECR
- EFFECR1
- EFFECR2
- EFFECR3
- EFFECR4

D'un autre côté, lorsqu'un module d'affichage est utilisé par un module utilisateur, seul un nombre bien déterminé de grilles d'écran intéresse ce module utilisateur. Ces mêmes grilles d'écran ne concernent souvent qu' un seul module utilisateur. De là l'idée de replacer le texte de programme correspondant à une grille d'écran particulière, dans la procédure ou le programme où elle est utilisée. Plutôt que de concentrer toutes les grilles d'écran dans une seule procédure d'affichage. Par exemple le menu M1 se retrouve dans le corps du programme principal.

Les modules de saisie de données ont subi le même sort. La saisie de données sur une grille d'écran particulière correspond à un texte de programme qui se retrouve avec le texte de cette grille d'écran. Souvent les deux aspects affichage et saisie ont été regroupés au sein d'une même procédure (par exemple SELEX). Lorsque plusieurs grilles d'écran semblables sont concernées par une même saisie, elles sont regroupées (par exemple SELPREPA, MESSAGE).

Le module d'affichage de valeurs dans une grille se retrouve également aux côtés du texte concernant la grille d'écran.

L' avantage de cette structure par rapport à une solution qui consiste à faire d'un module une procédure, est d'épargner un grand nombre de paramètres. De plus le segment PREPARATION se trouve déchargé du traitement des grilles d'écran qui concerne le segment EXECUTION et vice versa.

3.1.3. Niveau 4.

Le niveau 4 comprend les modules

- état fichier.
- sélection préparation.
- une création.
- une modification.
- une consultation.

La procédure EXAFICH correspond au module état fichier. Cette procédure invoque uniquement des procédures d'accès fichiers et concerne aussi la partie exécution. Nous l'avons donc placée dans la UNIT FICHIER de la librairie système aux côtés des procédures d'accès fichiers.

Les procédures SELPREPA de la partie préparation et SELPREPABIS de la partie exécution correspondent au module de selection préparation. Cette scission a été opérée afin de ne pas charger la partie préparation, de code destiné à la partie exécution et vice versa.

Les modules une création, une consultation, une modification, sont indépendants les uns des autres. Ils ont été isolés en segments :

- segment procédure PREPARE.
- segment procédure AFFICHE.
- segment procédure MODIFIE.

Lors d'une création, une consultation ou une modification on travaille sur les mêmes grilles d'écran. Cette segmentation a donc pour effet de grossir le fichier code puisque chaque texte de programme traitant d'une grille d'écran est reproduit trois fois. Cependant, la place mémoire sur la disquette n'est pas une ressource critique ,contrairement à l'espace en mémoire centrale.

3.1.4. Le niveau 5.

Le niveau 5 comprend les modules :

- session création.
- session modification.
- session consultation.
- session suppression.
- session liste des préparations existantes.

L'ensemble de ces modules correspond au corps de la segment procedure PREPARATION.

3.2. Implementation de la partie exécution.

La partie génération des exercices n'ayant pas été implémentée, nous nous limitons à la présentation de la structure de l'interface graphique. Néanmoins, nous formulerons une proposition de structure pour la partie génération.

La segment procédure EXECUTION se compose de :

- l'interface graphique.
- la segment procédure SESSION1.
- la segment procédure SESSION2.
- la segment procédure SESSION3.
- un ensemble de procédures :
  - SELEXBIS.
  - SELPREPABIS.
  - RESULTAT.

3.2.1. L'interface graphique.

3.2.1.1. Le niveau 2.

Les modules outils sont déjà implémentés et disponibles dans la librairie SYSTEM.LIBRARY.

3.2.1.2. Le niveau 3.

A chaque module du niveau 3, correspond une procédure qui porte le même nom et utilise les mêmes paramètres. Afin de rendre ces procédures facilement accessibles et pour permettre leur utilisation dans un autre contexte, nous les avons rassemblées dans une REGULAR UNIT apelée NIVEAU1. Nous avons préféré la REGULAR UNIT à l'INTRINSIC UNIT, afin de libérer la mémoire de l'interface graphique lorsqu'il n'est pas nécessaire.

3.2.1.3. Le niveau 4.

Chaque module a donné naissance à une procédure. Le nom et les paramètres de la procédure correspondent à ceux du module. Toutes ces procédures sont rassemblées dans la REGULAR UNIT NIVEAU2 qui utilise NIVEAU1. Ces procédures jouissent donc des deux propriétés accessibilité et réutilisabilité, au même titre que les autres modules outils.

3.2.2. Génération des exercices.

3.2.2.1. Le niveau 2.

Les modules outils sont disponibles dans la librairie SYSTEM.LIBRARY. L'interface graphique est disponible dans la librairie GRAPH.LIBRARY.

3.2.2.2. Le niveau 3.

Le niveau 3 comprend les modules :

- dessin.
- gestion réponse1.
- gestion réponse2.
- génération aléatoire.
- selprepa.
- accès fichier.
- gestion d'écran.

Les modules de gestion d'écran correspondent aux procédures SELPREPABIS, SELEXBIS et RESULTAT. Ces procédures s'occupent de l'affichage et de la saisie réponse des grilles d'écran (M8, P13, P14, P15, P16 et P38).

Les modules accès fichiers sont contenus dans la UNIT FICHIER disponible dans la librairie SYSTEM.LIBRARY.

Le module génération aléatoire correspondra à une procédure qui sera placée dans une REGULAR UNIT, afin de la rendre facilement accessible à tous les niveaux.

Les modules de dessin et de gestion réponse deviendront des procédures locales aux segments procédures qui les invoquent.

3.2.2.3. Le niveau 4.

Le niveau 4 comprend les modules :

- générer situation.
- générer distorsion.

Ces modules deviendront des procédures qui seront intégrées localement aux segments procédures.

3.2.2.4. Le niveau 5.

Le niveau 5 comprend les modules :

- session exercice1.
- session exercice2.
- session exercice3.

Les sessions d'exercices étant totalement indépendantes les unes des autres, chaque session d'exercices deviendra une segment procedure (SESSION1, SESSION2, SESSION3).

CONCLUSION

Nous avons réparti nos conclusions sous deux rubriques, selon qu'elles formulent des réflexions sur le contexte du travail, ou qu'elles traitent de la réalisation du didacticiel.

1. Le contexte.

Pour réaliser ce travail, nous avons pris certaines options concernant les auteurs, l'enseignement, le type de didacticiel et le matériel. Les choix du niveau de l'enseignement visé et du sujet du didacticiel nous semblent peu importants en ce sens que nous aurions pu tout aussi bien entreprendre un didacticiel traitant de géographie pour l'enseignement secondaire. Nous livrons ici quelques réflexions relatives aux choix des auteurs et du matériel.

Les auteurs.

Nous avons opté pour la collaboration enseignant-informaticien, l'un apportant ses connaissances en pédagogie, l'autre en informatique. Cette formule de travail, même si elle requiert un grand investissement de temps, nous semble tout à fait adéquate et particulièrement enrichissante pour les deux parties. Elle permet de mettre à profit les compétences de chacun, et oblige en outre à formuler complètement et clairement ses idées.

La condition essentielle d'un travail fructueux est l'établissement d'un contrat de confiance entre les partenaires. Chacun doit reconnaitre les limites de ses compétences et faire confiance à l'autre lorsqu'il sort de ces limites. Nous avons expérimenté ce mode de travail lors des étapes d'élaboration du projet et d'analyse fonctionnelle.

Enseignant et informaticien contribuent également à l'élaboration du projet. Le projet est travaillé à un niveau pédagogique par l'enseignant et en termes de faisabilité pour l'informaticien. Si l'enseignant ne possède aucune formation en informatique, l'informaticien devra intervenir pour aider à tirer le meilleur parti des avantages de l'ordinateur comme outil d'aide à l'enseignement.

CONCLUSION

L'analyse fonctionnelle consiste en une étude plus fouillée de tous les concepts : on précise les questions de détail, on envisage les cas particuliers... . Il s'agit d'un travail de rigueur et de précision auquel l'enseignant est peu habitué. L'informaticien tient donc le rôle de l'interrogateur. Les discussions se fondent souvent sur des exemples précis. C'est à l'informaticien qu'il revient de généraliser les concepts, de les résumer et les reproposer à la discussion. Ce genre de travail n'est pas familier à l'enseignant. La principale difficulté rencontrée par l'enseignant lors de cette étape est de se faire une représentation dynamique du didacticiel. La solution de la réalisation de la maquette nous paraît excellente . Elle constitue un support beaucoup plus parlant que la feuille de papier. De plus, elle permet à l'informaticien de se familiariser avec son outil de travail.

Le matériel.

Nous avons opté pour le micro-ordinateur APPLE 2. Le choix du micro-ordinateur de par son prix et son caractère poste de travail individuel nous parait excellent. Le langage choisi pour développer le didacticiel a été le Pascal UCSD disponible sur Apple. Nous ne remettons pas en cause ce choix, cependant nous émettons deux critiques à cet égard.

La première concerne la documentation Pascal Apple. Cette documentation a la réputation d'être de bonne qualité et elle la mérite, cependant :

- L'utilisation de la couleur pour le dessin de traits fins est autorisée en théorie, mais sur le matériel couleur dont nous disposons, elle ne donne pas les résultats attendus.
- Nous n'avons trouvé aucune documentation se rapportant au chaînage de programmes alors que nous connaissons l'existence de cette possibilité.

La seconde remarque porte sur les librairies du système. Lorsqu'une seule routine d'une librairie est utilisée, la librairie entière doit être chargée, surchargeant ainsi la mémoire de nombreuses autres routines complètement inutiles. Il serait donc intéressant de réaliser un système où l'utilisateur pourrait se construire ses propres librairies à partir de routines mises à sa disposition et selon ses besoins.

2. Le didacticiel.

Après un bref résumé de la situation de notre travail, nous examinerons successivement les faiblesses et les points forts du didacticiel. Le troisième point sera consacré aux extensions envisageables.

Situation du travail.

Rappelons que notre objectif de départ était la réalisation d'un didacticiel visant à améliorer la compréhension des opérations arithmétiques, dans le cas de la structure additive. Pour ce faire, nous avons imaginé une série d'exercices de reconnaissance. Le didacticiel est constitué de deux parties : une partie préparation des exercices, destinée à l'enseignant et une partie exécution des exercices, destinée à l'élève. La partie exécution des exercices est elle-même composée d'un interface graphique et d'une partie génération des exercices.

Nous avons mené ce projet complètement jusqu'à l'étape d'analyse organique. Nous avons également terminé l'implémentation de la partie préparation des exercices et de l'interface graphique. L'implémentation de la génération des exercices reste afaire.

Une des grandes satisfactions que ce travail nous a procuré est d'avoir permis de s'occuper du projet depuis sa conception au travers des différentes étapes de la vie d'un projet jusqu'à l'étape d'implémentation. Nous regrettons cependant de n'avoir pas abordé la phase d'expérimentation du produit.

Les points faibles.

Les deux grands points faibles de ce didacticiel se situent au niveau de la remédiation et du suivi des élèves.

Nous n'avons envisagé aucun processus d' interprétation des erreurs des élèves et encore moins de remédiation. Pour entreprendre un tel travail il faudrait disposer d'une typologie des erreurs fréquentes des élèves.

Nous n'avons pas envisagé la création de fichiers d'élèves qui permettraient de retenir différentes informations qui peuvent être utiles à l'enseignant telles que les résultats obtenus ou les types d'erreurs commises. Ces informations permettent par exemple de suivre les progrès des élèves ou de repère des notions qui n'aurait pas été comprises par l'ensemble de la classe. Le contenu de tels fichiers d'élèves est à définir selon les besoins de l'enseignant.

CONCLUSION

Les points forts.

La force de ce didacticiel réside dans son caractère paramétrable et aléatoire. La paramétrisation a été établie suivant le principe "qui peut le plus peut le moins" . Elle permet non seulement d'adapter les exercices aux élèves ,mais aussi d'élargir le public de départ, celui de la deuxième année, à d' autres années du primaire. La génération aléatoire des exercices permet d'éviter la monotonie et ne limite donc pas la fréquence d'utilisation. Souplesse et diversité sont donc les principales qualités de ce didacticiel.

Nous soulignerons également l'effort fourni au niveau de l'ergonomie. Tant dans la partie exécution que préparation des exercices, nous avons apporté un soin particulier à la réalisation des graphiques et des grilles d'écrans. Nous avons travaillé la standardisation des écrans et des modes de réponses ainsi que le contrôle de validité des données.

Les extensions.

Les premières extensions envisageables se proposent par rapport aux lacunes existantes, à savoir travailler la remédiation et le suivi des élèves.

Une autre extension intéressante consisterait à améliorer l'adaptibilité des exercices au niveau des élèves. Dans le système actuel, l'adaptation peut se faire par paramétrisation des exercices à l'aide des préparations. On pourrait y ajouter une processus de chaînage des préparations. Il permettrait de déterminer, après exécution d'une préparation et en fonction des résultats obtenus, quelle est la préparation suivante à exécuter. Ce chainage pourrait être défini par le biais de tables de décisions. A chaque préparation serait associée une table de décision qui décrirait les préparations à exécuter en fonction des résultats obtenus. Ces tables de décisions devraient être établies par l'enseignant.

Enfin, nous serions heureux que ce travail puisse susciter la réalisation d'autres didacticiel portant sur des exercices d'imitation, d'ajustement ou d'achèvement de l'initiative.

BIBLIOGRAPHIE.

[APP;80] Apple 2 :
" Apple Pascal
Language Reference Manual ".
Apple Computer inc., 1980.

[BER;82] Berleur, J. et al. :
" Une Societè Informatisée
Pour qui? Pour quoi? Comment? ".
Presse Universitaire de Namur,
Mai 1982, pp. 375-413.

[BER;84] Berleur, J. et al. :
" L'Appropiation Sociale de l'Informatique à... ".
Presse Universitaire de Namur,
Août Septembre 1984, pp. 117-229.

[BES;82] Bestougeff, H. et Fargette, J.P. :
" Enseignement et Ordinateur ".
Cedic/Fernand Nathan, 1982.

[BOS;83] Bossuet, G. :
" L'Ordinateur à l'Ecole ".
Presse Universitaire de France, 1982.

[CAN;82] Cannella, R. et al. :
" Dossier Enseignement ".
L'Ordinateur Individuel,
Octobre 1982, pp. 101-127.

[COL;83] Colibri, J. :
" Découvrez Pascal sur Apple 2 ".
Nmemodyne, Mars 1983.

[CON;80] Conseil Central de l'Enseignement Primaire Catholique :
" Programmes des Etudes pour les Ecoles
Primaires Catholique, Mathématiques ".
Enseignement National Catholique, 1980.

[DIR;77] Direction Générale de l'Organisation des Etudes :
" Commentaire des Progammes de Mathématiques
pour les trois cycles, 4-A ".
Réforme Pédagogique de l'Enseignement Primaire, 1977.

[DIR;80] Direction Générale de l'Organisation des Etudes :
" Commentaire des Progammes de Mathématiques
pour les trois cycles, 4-D ".
Réforme Pédagogique de l'Enseignement Primaire, 1980.

[DUC;83] Duchâteau, CH. :
" Informatique et Enseignement Secondaire
une Proposition de Bibliographie ".
Cefis, Faculté N.D. de la Paix,
Namur, Mai 1983.

BIBLIOGRAPHIE.

[FER;83] Ferber, J. :
" Dossier Intelligence Artificielle et Logo ".
Micro-Système, Avril 1983, pp. 87-100.

[GRA;83] Grant, CH. W. et Butah, J. :
" Introduction au P-Système UCSD ".
Sybex, 1983.

[HAB;80] Habran, L. et Linsingh, C. :
" Mathématiques Moderne
Livre du Maître no. 1 ".
Labor, collection W. Servais, 1980.

[HAS;82] Hasquin, H. et al. :
" Enseignement Assisté par Ordinateur ".
Bulletin de l'Université Libre de Bruxelles
et de l'Union des Anciens Etudiants,
Septembre 1982.

[HYP;82] Hypothèse d'Ecole :
" L'Ordinateur à l'Ecole ".
Journal de classe, Septembre Octobre 1982.

[JOB;83] Jobert, G. et Perriault, J. :
" L'Enseignement Assisté par Ordinateur ".
Education Permanante, Université Paris-Dauphine,
Decembre 1983.

[LET;81] Letourneux, J.P. :
" Dossier Et l'Informatique... ".
Cahier Pédagogique,
Novembre 1981.

[MYE;75] Myers, G.J. :
" Reliable Software through Composite Design ".
Petrocelli/Charter, New York, 1975.

[PAP;81] Papert, S. :
" Le Jaillissement de l'Esprit
Ordinateurs et Apprentissage ".
Flammarion, Paris, 1981.

[PEU;83] Peuchot, M. :
" L'Enseignement Assisté par Ordinateur ".
Micro-Système, Janvier Février 1982, pp. 87-99.

[SCH;81] Schartz, B. :
" L'Informatique et l'Education,
Rapport à la CEE ".
La Documentation Française, Avril 1981.

BIBLIOGRAPHIE.

[SIM;81] Simon, J.C. :
" L'Education et l'Informatisation de la Société, rapport au Président de la République ".
Payard, la Documentation Française, 1981.

[TOR;83] Tortel, C. et al. :
" Dossier L'Ordinateur Outil Pédagogique ".
L'Ordinateur Individuel,
Octobre 1983, pp. 132-157.

[WAR;84] Warusfel, A. et al. :
" Dossier Enseignement et Informatique ".
L'Ordinateur Individuel,
Octobre 1984, pp. 110-123.

Facultès Universitaire Notre-Dame de la Paix (Namur)

Institut d'Informatique

*** ANNEXES ***

E . A . O.

PROJET D'AIDE A LA COMPREHENSION DES OPERATIONS ARITHMETIQUES POUR L'ENSEIGNEMENT PRIMAIRE.

Mèmoire presentè par

Catherine TONNEAU

en vue de l'obtention
du titre de
licencié et maître en informatique

Promoteur Claude CHERTON.

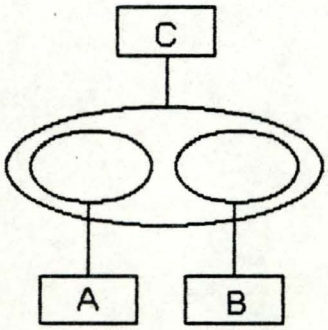
Annèe academique 1984 - 1985

annexe 1 : dessins des représentations.

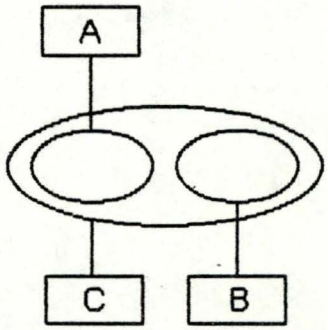
Diagrammes.

forme 1 | forme 2 | forme 3

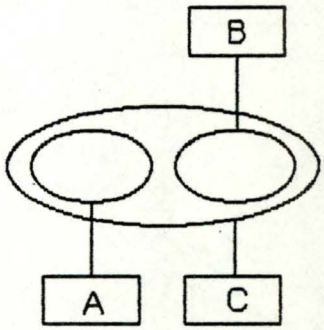
nord



D1

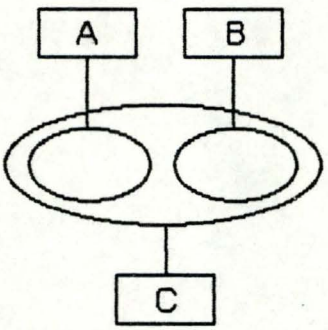


D2

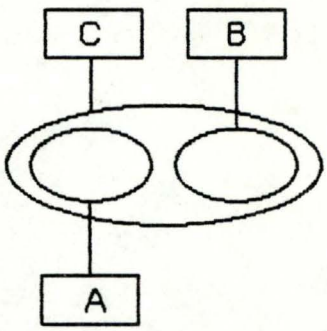


D3

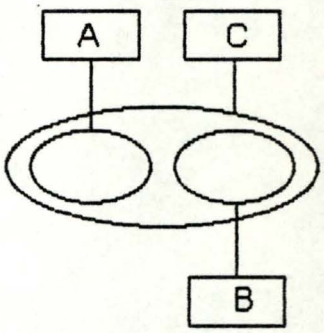
sud



D4



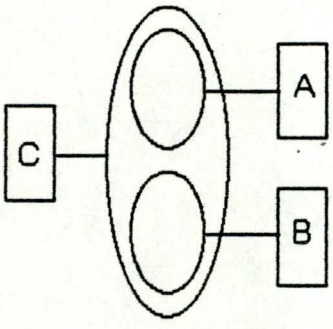
D5



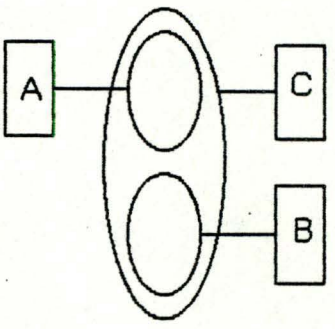
D6

forme 1 forme 2 forme 3

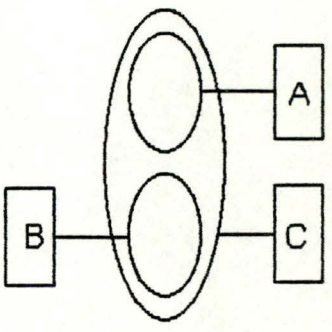
ouest



D7

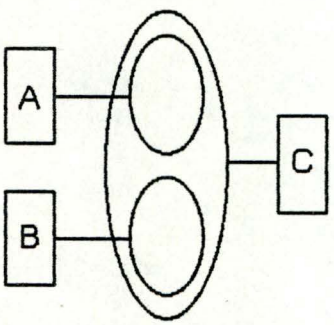


D8

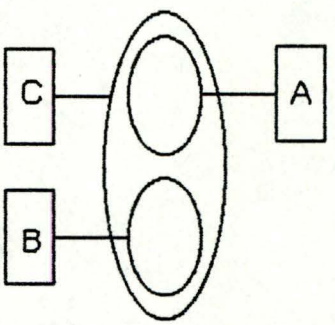


D9

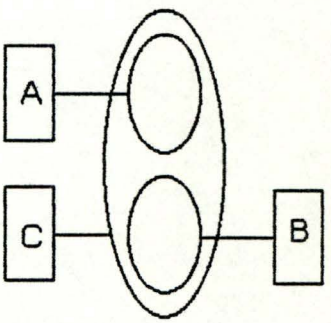
est



D10



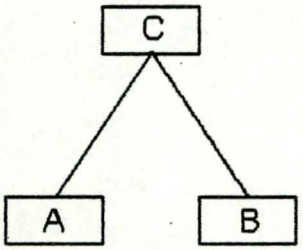
D11



D12

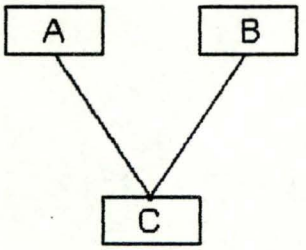
Arbres.

nord



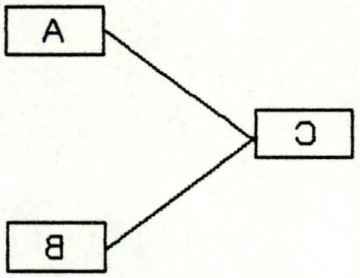
A1

sud



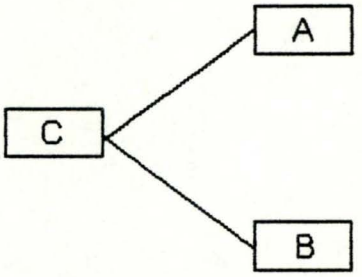
A2

est



A3

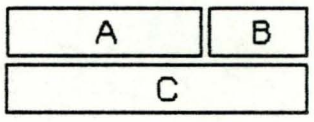
ouest



A4

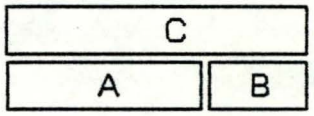
Lineaires.

nord



L1

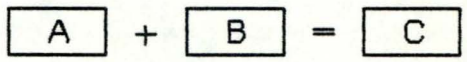
sud



L2

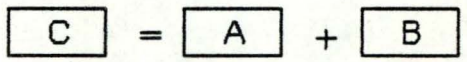
Ecriture arithmetique du signe +

nord



EA+1

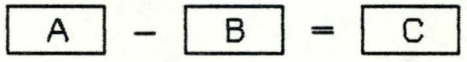
sud



EA+2

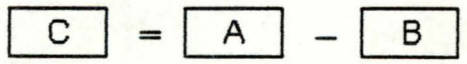
Ecriture arithmetique du signe −

nord



EA−1

sud



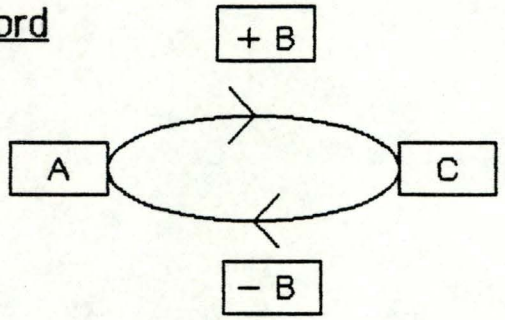
EA−2

Graphes

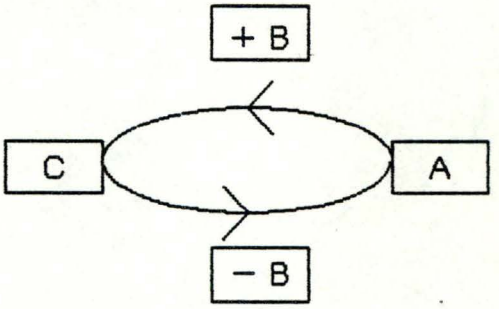
Forme1

Forme2

nord

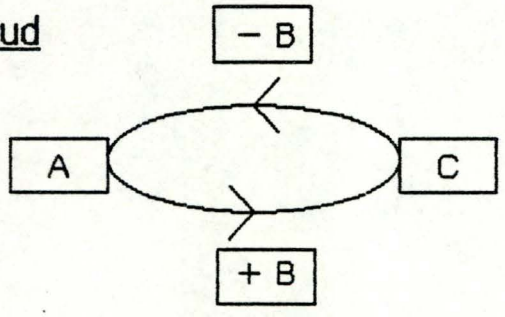


G1

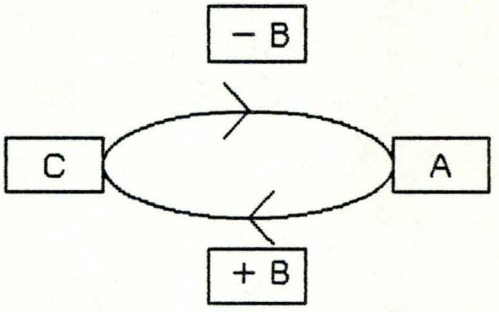


G2

sud

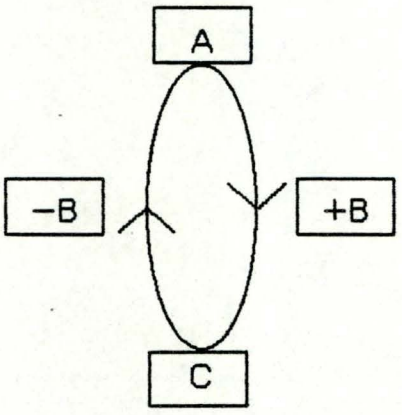


G3

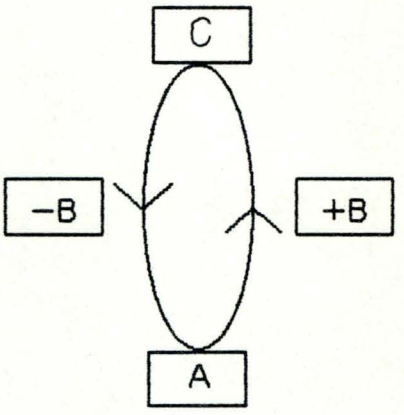


G4

est

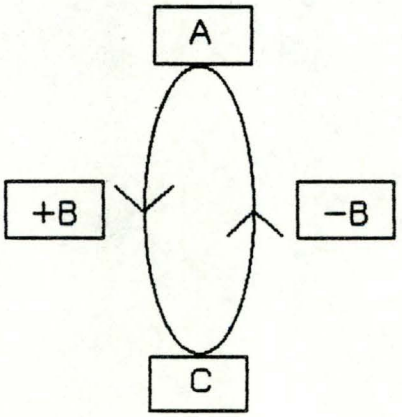


G5

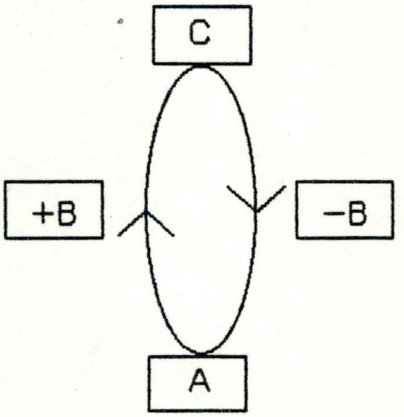


G6

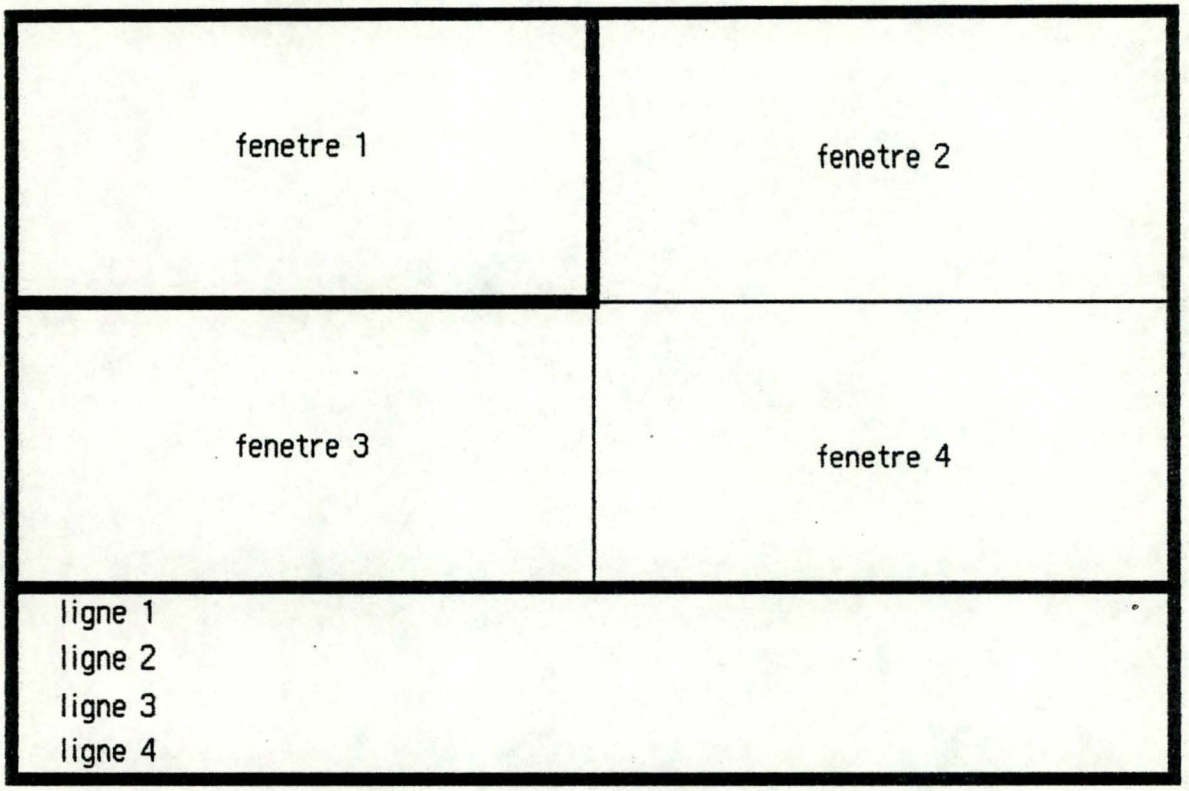
ouest



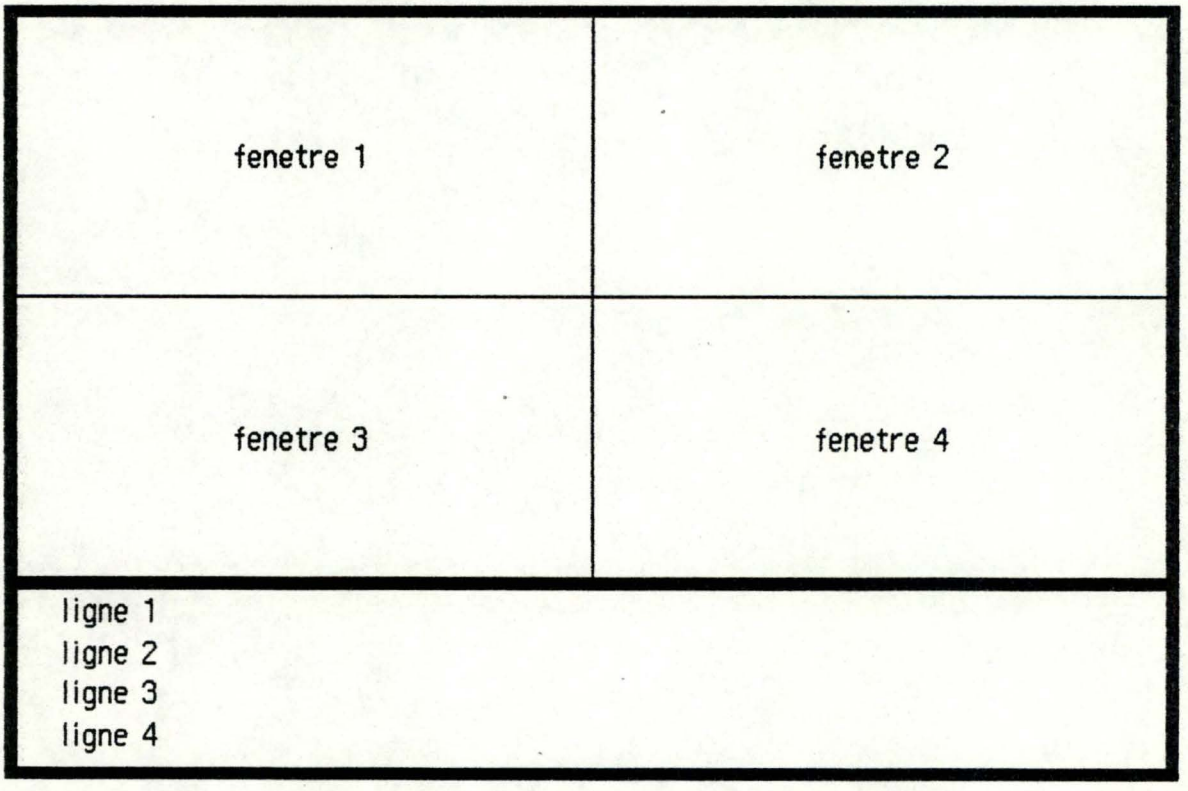
G7



G8



SHEMA 1



SHEMA 2

annexe 2 : Les grilles d'écran.

description DES GRILLES D'ECRAN.

1) Les menus.

```
   0         1         2         3
   01234567890123456789012345678901234567890
  ------------------------------------------
 0         *********************************
 1         *                               *
 2         * STRUCTURE ADDITIVE :          *
 3         *                               *
 4         * EXERCICES DE RECONNAISSANCE   *
 5         *                               *
 6         *********************************
 7
 8
 9    MENU GENERAL
10    ============
11
12                  1- PREPARATION
13
14                  2- EXECUTION
15
16
17
18     VOTRE CHOIX :-
19     ------------
20
21
22
23
  ------------------------------------------

   M1
```

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1            MENU PREPARATION
 2            ================
 3
 4     1-LISTER LES PREPARATIONS EXISTANTES
 5
 6     2-AFFICHER UNE PREPARATION
 7
 8     3-CREER UNE PREPARATION
 9
10     4-MODIFIER UNE PREPARATION
11
12     5-SUPPRIMER UNE PREPARATION
13
14     6-FIN DE PREPARATION
15
16
17     VOTRE CHOIX : -
18     ------------
19
20
21
22
23
  ------------------------------------------------------

    M2

   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0       LISTER LES PREPARATIONS EXISTANTES
 1       ==================================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4
 5
 6 1-SELECTION PARMI 3 SITUATIONS EXPRIMEES
 7   AVEC LA MEME REPRESENTATION
 8
 9 2-SELECTION PARMI 3 SITUATIONS EXPRIMEES
10   AVEC DES REPRESENTATIONS DIFFERENTES
11
12 3-ASSOCIATION DES SITUATIONS SEMBLABLES
13
14
15
16 VOTRE CHOIX :_
17 ------------
18
19
20
21
22
23
  ------------------------------------------------------

    M3
```

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0          AFFICHER UNE PREPARATION
 1          ========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4
 5
 6 1-SELECTION PARMI 3 SITUATIONS EXPRIMEES
 7   AVEC LA MEME REPRESENTATION
 8
 9 2-SELECTION PARMI 3 SITUATIONS EXPRIMEES
10   AVEC DES REPRESENTATIONS DIFFERENTES
11
12 3-ASSOCIATION DES SITUATIONS SEMBLABLES
13
14
15
16 VOTRE CHOIX :-
17 ------------
18
19
20
21
22
23
  ------------------------------------------------------
```

M4

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0            CREER UNE PREPARATION
 1            =====================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4
 5
 6 1-SELECTION PARMI 3 SITUATIONS EXPRIMEES
 7   AVEC LA MEME REPRESENTATION
 8
 9 2-SELECTION PARMI 3 SITUATIONS EXPRIMEES
10   AVEC DES REPRESENTATIONS DIFFERENTES
11
12 3-ASSOCIATION DES SITUATIONS SEMBLABLES
13
14
15
16 VOTRE CHOIX :-
17 ------------
18
19
20
21
22
23
  ------------------------------------------------------
```

M5

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0          MODIFIER UNE PREPARATION
 1          ========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4
 5
 6 1-SELECTION PARMI 3 SITUATIONS EXPRIMEES
 7   AVEC LA MEME REPRESENTATION
 8
 9 2-SELECTION PARMI 3 SITUATIONS EXPRIMEES
10   AVEC DES REPRESENTATIONS DIFFERENTES
11
12 3-ASSOCIATION DES SITUATIONS SEMBLABLES
13
14
15
16 VOTRE CHOIX :-
17 ----------
18
19
20
21
22
23
  ------------------------------------------------------
```

M6

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0          SUPPRIMER UNE PREPARATION
 1          =========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4
 5
 6 1-SELECTION PARMI 3 SITUATIONS EXPRIMEES
 7   AVEC LA MEME REPRESENTATION
 8
 9 2-SELECTION PARMI 3 SITUATIONS EXPRIMEES
10   AVEC DES REPRESENTATIONS DIFFERENTES
11
12 3-ASSOCIATION DES SITUATIONS SEMBLABLES
13
14
15
16 VOTRE CHOIX :-
17 ----------
18
19
20
21
22
23
  ------------------------------------------------------
```

M7

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1            EXECUTION DES EXERCICES
 2            ======================
 3
 4 EXERCICES DE RECONNAISSANCE :
 5
 6
 7 1-SELECTION PARMI 3 SITUATIONS EXPRIMEES
 8   AVEC LA MEME REPRESENTATION
 9
10 2-SELECTION PARMI 3 SITUATIONS EXPRIMEES
11   AVEC DES REPRESENTATIONS DIFFERENTES
12
13 3-ASSOCIATION DES SITUATIONS SEMBLABLES
14
15
16 VOTRE CHOIX :-
17 ------------
18
19
20
21
22
23
  ------------------------------------------------------
```

M8

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1          MODIFIER UNE PREPARATION
 2          ========================
 3
 4
 5 1- ENREGISTREMENT DES MODIFICATIONS
 6    ----------------
 7
 8
 9 2- ANNULATION DES MODIFICATIONS
10    ----------
11
12
13
14 VOTRE CHOIX :-
15 ------------
16
17
18
19
20
21
22
23
  ------------------------------------------------------
```

2) Les messages.

```
          0         1         2         3
   01234567890123456789012345678901234567890123456789
  ----------------------------------------------------
 0
 1
 2
 3
 4
 5
 6
 7
 8
 9
10 CREATION IMPOSSIBLE :
11
12 10 PREPARATIONS ONT DEJA ETE CREES
13
14
15
16
17
18
19
20
21
22
23
  ----------------------------------------------------
```

MSG1

```
          0         1         2         3
   01234567890123456789012345678901234567890123456789
  ----------------------------------------------------
 0
 1
 2
 3
 4
 5
 6
 7
 8
 9
10 CREATION INTERROMPUE :
11
12 AUCUNE INFORMATION N EST ENREGISTREE
13
14
15
16
17
18
19
20
21
22
23
  ----------------------------------------------------
```

MSG2

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3
 4
 5
 6
 7
 8
 9
10 CREATION TERMINEE :
11
12 LA CREATION A BIEN ETE ENREGISTREE
13
14
15
16
17
18
19
20
21
22
23
  ------------------------------------------------------
```

MSG3

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3
 4
 5
 6
 7
 8
 9
10 SUPPRESSION IMPOSSIBLE :
11
12 PAS DE PREPARATION A SUPPRIMER
13
14
15
16
17
18
19
20
21
22
23
  ------------------------------------------------------
```

MSG4

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3
 4
 5
 6
 7
 8
 9
10 SUPPRESSION TERMINEE :
11
12 LA PREPARATTION A BIEN ETE SUPPRIMEE
13
14
15
16
17
18
19
20
21
22
23
  ------------------------------------------------------
```

MSG5

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3
 4
 5
 6
 7
 8
 9
10 MODIFICATION IMPOSSIBLE :
11
12 PAS DE PREPARATION A MODIFIER
13
14
15
16
17
18
19
20
21
22
23
  ------------------------------------------------------
```

MSG6

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3
 4
 5
 6
 7
 8
 9
10 MODIFICATION TERMINEE :
11
12 LES MODIFICATIONS ONT ETE ENREGISTREES
13
14
15
16
17
18
19
20
21
22
23
  ------------------------------------------------------
```

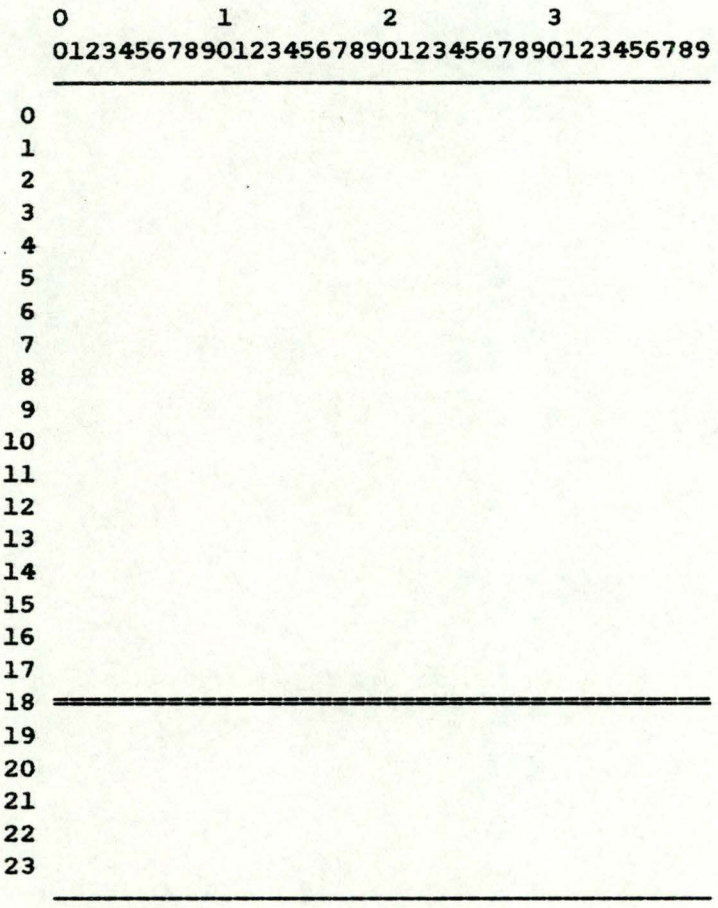
MSG7

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3
 4
 5
 6
 7
 8
 9
10 MODIFICATION TERMINEE :
11
12 AUCUNE MODIFICATION N EST ENREGISTREE
13
14
15
16
17
18
19
20
21
22
23
  ------------------------------------------------------
```

MSG8

3) Les fenêtres.

3.1) Initialisation.



3.2) Les fenêtres de présentation.

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0      LISTE DES PREPARATIONS EXISTANTES
 1      ================================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 5 AVEC LA MEME REPRESENTATION
 6
 7 0-PREPARATION PAR DEFAUT
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------
```

P1

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0      LISTE DES PREPARATIONS EXISTANTES
 1      ================================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 5 AVEC DES REPRESENTATIONS DIFFERENTES
 6
 7 0-PREPARATION PAR DEFAUT
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------
```

P2

```
   0         1         2         3
   01234567890123456789012345678901234567890
  ------------------------------------------
 0     LISTE DES PREPARATIONS EXISTANTES
 1     ==================================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 ASSOCIATION DES SITUATIONS
 5
 6
 7 0-PREPARATION PAR DEFAUT
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------
```

P3

```
   0         1         2         3
   01234567890123456789012345678901234567890
  ------------------------------------------
 0          AFFICHER UNE PREPARATION
 1          ========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 5 AVEC LA MEME REPRESENTATION
 6
 7 0-PREPARATION PAR DEFAUT
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------
```

P4

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0          AFFICHER UNE PREPARATION
 1          ========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 5 AVEC DES REPRESENTATIONS DIFFERENTES
 6
 7 0-PREPARATION PAR DEFAUT
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------
```

P5

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0          AFFICHER UNE PREPARATION
 1          ========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 ASSOCIATION DES SITUATIONS
 5
 6
 7 0-PREPARATION PAR DEFAUT
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------
```

P6

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1          MODIFIER UNE PREPARATION
 2          ========================
 3
 4 EXERCICES DE RECONNAISSANCE :
 5 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 6 AVEC LA MEME REPRESENTATION
 7
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------
```

P7

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1          MODIFIER UNE PREPARATION
 2          ========================
 3
 4 EXERCICES DE RECONNAISSANCE :
 5 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 6 AVEC DES REPRESENTATIONS DIFFERENTES
 7
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------
```

P8

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1          MODIFIER UNE PREPARATION
 2          ========================
 3
 4 EXERCICES DE RECONNAISSANCE :
 5 ASSOCIATION DES SITUATIONS
 6
 7
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------
```

P9

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1          SUPPRIMER UNE PREPARATION
 2          =========================
 3
 4 EXERCICES DE RECONNAISSANCE :
 5 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 6 AVEC LA MEME REPRESENTATION
 7
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------
```

P10

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------------
 0          SUPPRIMER UNE PREPARATION
 1          ==========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 5 AVEC DES REPRESENTATIONS DIFFERENTES
 6
 7 1-
 8 2-
 9 3-
10 4-
11 5-
12 6-
13 7-
14 8-
15 9-
16
17
   ------------------------------------------------------
```

P11

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------------
 0          SUPPRIMER UNE PREPARATION
 1          ==========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 ASSOCIATION DES SITUATIONS
 5
 6
 7 1-
 8 2-
 9 3-
10 4-
11 5-
12 6-
13 7-
14 8-
15 9-
16
17
   ------------------------------------------------------
```

P12

```
   0         1         2         3
   01234567890123456789012345678901234567890
   ----------------------------------------
 0          EXECUTER UNE PREPARATION
 1          ========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 5 AVEC LA MEME REPRESENTATION
 6
 7 0-PREPARATION PAR DEFAUT
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
   ----------------------------------------

   P13
```

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0          EXECUTER UNE PREPARATION
 1          ========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 SELECTION PARMI 3 SITUATIONS EXPRIMEES
 5 AVEC DES REPRESENTATIONS DIFFERENTES
 6
 7 0-PREPARATION PAR DEFAUT
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------

   P14
```

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0          EXECUTER UNE PREPARATION
 1          ========================
 2
 3 EXERCICES DE RECONNAISSANCE :
 4 ASSOCIATION DES SITUATIONS
 5
 6
 7 0-PREPARATION PAR DEFAUT
 8 1-
 9 2-
10 3-
11 4-
12 5-
13 6-
14 7-
15 8-
16 9-
17
  ------------------------------------------------------

   P15
```

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0    AFFICHAGE : ----------------------------
 1    =========
 2
 3
 4
 5
 6
 7
 8
 9
10
11
12
13
14
15
16
17
  ------------------------------------------------------
```

P16

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0    CREATION : ----------------------------
 1    ========
 2
 3
 4
 5
 6
 7
 8
 9
10
11
12
13
14
15
16
17
  ------------------------------------------------------
```

P17

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0 MODIFICATION : ------------------------
 1 ===========
 2
 3
 4
 5
 6
 7
 8
 9
10
11
12
13
14
15
16
17

   ------------------------------------------------
```

P18

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0
 1
 2
 3 REPRESENTATIONS UTILISEES :
 4 --------------------------
 5 1-DIAGRAMME ------> --
 6
 7 2-GRAPHE ---------> --
 8
 9 3-ARBRE ----------> --
10
11 4-LINEAIRE ------> --
12
13 5-MATH.+. -------> --
14
15 6-MATH.-. -------> --
16
17

   ------------------------------------------------
```

P19

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0
 1
 2
 3 DISPOSITION SPATIALE VARIABLE :
 4 ------------------------------
 5 1-DIAGRAMME ------> --
 6
 7 2-GRAPHE ---------> --
 8
 9 3-ARBRE ----------> --
10
11 4-LINEAIRE ------> --
12
13 5-MATH.+. -------> --
14
15 6-MATH.-. -------> --
16
17
   ------------------------------------------------
```

P20

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0
 1
 2
 3 DESSIN VARIABLE :
 4 ----------------
 5 1-DIAGRAMME ------> --
 6
 7 2-GRAPHE ---------> --
 8
 9
10
11
12
13
14
15
16
17
   ------------------------------------------------
```

P21

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0
 1
 2
 3 NOMBRE D'EXERCICES :
 4 ------------------
 5 SOIT UNE SITUATION DE DEPART PRESENTEE :
 6 SOUS FORME DE DIAGRAMME
 7   ----------
 8 ET UNE SITUATION A RECONNAITRE
 9 SOUS FORME DE :                    NBR.EX.:
10        ---------
11       1-DIAGRAMME------------->   --
12       2-GRAPHE---------------->   --
13       3-ARBRE----------------->   --
14       4-LINEAIRE-------------->   --
15         5-MATH.+.--------------->   --
16        6-MATH.-.--------------->   --
17
   ------------------------------------------------
```

P22

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0
 1
 2
 3 NOMBRE D'EXERCICES :
 4 --------------------
 5 SOIT UNE SITUATION DE DEPART PRESENTEE :
 6 SOUS FORME DE GRAPHE
 7   -------
 8 ET UNE SITUATION A RECONNAITRE
 9 SOUS FORME DE :                    NBR.EX.:
10        ---------
11       1-DIAGRAMME------------->   --
12       2-GRAPHE---------------->   --
13       3-ARBRE----------------->   --
14       4-LINEAIRE-------------->   --
15         5-MATH.+.--------------->   --
16        6-MATH.-.--------------->   --
17
   ------------------------------------------------
```

P23

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ----------------------------------------------------
 0
 1
 2
 3 NOMBRE D'EXERCICES :
 4 --------------------
 5 SOIT UNE SITUATION DE DEPART PRESENTEE :
 6 SOUS FORME DE ARBRE
 7   -----
 8 ET UNE SITUATION A RECONNAITRE
 9 SOUS FORME DE :                    NBR.EX.:
10       ---------
11       1-DIAGRAMME------------->  --
12       2-GRAPHE---------------->  --
13       3-ARBRE----------------->  --
14       4-LINEAIRE-------------->  --
15         5-MATH.+.--------------->  --
16        6-MATH.-.--------------->  --
17
   ----------------------------------------------------
```

P24

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ----------------------------------------------------
 0
 1
 2
 3 NOMBRE D'EXERCICES :
 4 --------------------
 5 SOIT UNE SITUATION DE DEPART PRESENTEE :
 6 SOUS FORME DE LINEAIRE
 7   ---------
 8 ET UNE SITUATION A RECONNAITRE
 9 SOUS FORME DE :                    NBR.EX.:
10       ---------
11       1-DIAGRAMME------------->  --
12       2-GRAPHE---------------->  --
13       3-ARBRE----------------->  --
14       4-LINEAIRE-------------->  --
15         5-MATH.+.--------------->  --
16        6-MATH.-.--------------->  --
17
   ----------------------------------------------------
```

P25

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3 NOMBRE D'EXERCICES :
 4 --------------------
 5 SOIT UNE SITUATION DE DEPART PRESENTEE :
 6 SOUS FORME DE MATH.+.
 7   -------
 8 ET UNE SITUATION A RECONNAITRE
 9 SOUS FORME DE :                    NBR.EX.:
10       ---------
11      1-DIAGRAMME------------------>  --
12      2-GRAPHE---------------------->  --
13      3-ARBRE----------------------->  --
14      4-LINEAIRE-------------------->  --
15        5-MATH.+.--------------------->  --
16       6-MATH.-.--------------------->  --
17
```

P26

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3 NOMBRE D'EXERCICES :
 4 --------------------
 5 SOIT UNE SITUATION DE DEPART PRESENTEE :
 6 SOUS FORME DE MATH.-.
 7   -------
 8 ET UNE SITUATION A RECONNAITRE
 9 SOUS FORME DE :                    NBR.EX.:
10       ---------
11      1-DIAGRAMME------------------>  --
12      2-GRAPHE---------------------->  --
13      3-ARBRE----------------------->  --
14      4-LINEAIRE-------------------->  --
15        5-MATH.+.--------------------->  --
16       6-MATH.-.--------------------->  --
17
```

P27

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3 NOMBRE D'EXERCICES :
 4 ---------------------
 5
 6 POUR LES SITUATIONS LIEES PAR :
 7
 8 1- ASSOCIATION 2-1-1 ----------> --
 9
10 2- ASSOCIATION 2-2 ------------> --
11
12 3- ASSOCIATION 3-1 ------------> --
13
14 4- ASSOCIATION 4 --------------> --
15
16 5- AUCUNE ASSOCIATION ---------> --
17
   ------------------------------------------------------
```

P28

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
  ------------------------------------------------------
 0
 1
 2
 3 REPORT A LA DIZAINE :
 4 ---------------------
 5
 6 LES SITUATIONS OU LE CALCUL DE A + B
 7
 8 OCCASIONNENT UN REPORT A LA DIZAINE
 9
10 SONT ADMISES ----------> --
11
12
13
14
15
16
17
   ------------------------------------------------------
```

P29

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0
 1
 2
 3 VALEUR MAXIMALE DE LA SOMME
 4 ---------------------------
 5
 6 SOIT A + B = C. LA PLUS GRANDE VALEUR QUE
 7
 8 C PEUT ATTEINDRE EST ---
 9
10
11
12
13
14
15
16
17
   ------------------------------------------------
```

P30

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0
 1
 2
 3 INVERSION DES PARTIES :
 4 -----------------------
 5 1-DIAGRAMME ------> ---
 6
 7 2-GRAPHE ---------> ---
 8
 9 3-ARBRE ----------> ---
10
11 4-LINEAIRE ------> ---
12
13 5-MATH.+. -------> ---
14
15 6-MATH.-. -------> ---
16
17
   ------------------------------------------------
```

P31

```
  0         1         2         3
  01234567890123456789012345678901234567890123456789
 ---------------------------------------------------
 0
 1
 2
 3 DISTORSIONS PAR RAPPORT A A+B=? SI A+B=C
 4 ----------------------------------
 5
 6 -MEMES SYMBOLES :
 7
 8       1- A+?=B , ?+B=A ----------> ---
 9    (B>A)   (A>B)
10
11 -A OU B REMPLACE PAR C :
12
13       2- C+B=? , A+C=? ----------> ---
14
15       3- ?+B=C , A+?=C ----------> ---
16
17
 ---------------------------------------------------
```



```
  0         1         2         3
  01234567890123456789012345678901234567890123456789
 ---------------------------------------------------
 0
 1
 2
 3 DISTORSIONS PAR RAPPORT A A+B=? SI A+B=C
 4 ----------------------------------
 5
 6 -A OU B REMPLACE PAR X :
 7
 8       4- X+B=? , A+X=? ----------> ---
 9
10       5- ?+B=X , A+?=X !
11    (X>B)   (X>A) !
12    !----------> ---
13          ?+X=B , X+?=A !
14    (X<B)   (X<A) !
15
16
17
 ---------------------------------------------------
```

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ----------------------------------------------------
 0
 1
 2
 3 DISTORSIONS PAR RAPPORT A A+B=? SI A+B=C
 4 ----------------------------------
 5
 6 -A ET B REMPLACE PAR C ET X :
 7
 8     6- X+C=? ------------------------> ---
 9
10     7- ?+C=X  , X+?=C ---------------> ---
11        (X>C)    (C>X)
12
13 -A ET B REMPLACE PAR X ET Y (X>Y) :
14
15     8- X+Y=? ------------------------> ---
16
17     9- ?+Y=X ------------------------> ---

   ----------------------------------------------------
```

P34

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ----------------------------------------------------
 0
 1
 2
 3 DISTORSIONS PAR RAPPORT A A+?=C SI A+B=C
 4 ----------------------------------
 5
 6 -MEMES SYMBOLES :
 7
 8      1- A+C=? ------------------------> ---
 9
10 -A OU C REMPLACE PAR B :
11
12      2- C+B= ? , A+B=? ---------------> ---
13
14      3- ?+B=C , A+?=B , ?+B=A --> ---
15            (B>A)   (A>B)
16
17

   ----------------------------------------------------
```

P35

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0
 1
 2
 3 DISTORSIONS PAR RAPPORT A A+?=C SI A+B=C
 4 ---------------------------------------
 5
 6 -A OU C REMPLACE PAR X :
 7
 8     4- X+C=? , A+X=? ----------> ---
 9
10     5- ?+C=X , A+?=X !
11        (X>C)   (X>A) !
12   !----------> ---
13        ?+X=C , X+?=A !
14        (X<C)   (X<A) !
15
16
17
   ------------------------------------------------
```



```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------
 0
 1
 2
 3 DISTORSIONS PAR RAPPORT A A+?=C SI A+B=C
 4 ---------------------------------------
 5
 6 -A ET C REMPLACES PAR B ET X :
 7
 8 6- X+B=? ----------------------> ---
 9
10      7- ?+B=X , X+?=B ----------> ---
11    (X>B)   (B>X)
12
13 -A ET B REMPLACES PAR X ET Y (X>Y) :
14
15      8- X+Y=? ----------------------> ---
16
17      9- ?+Y=X ----------------------> ---
   ------------------------------------------------
```

```
  0         1         2         3
  01234567890123456789012345678901234567890
  ----------------------------------------
0             RESULTAT DE LA SESSION
1             ======================
2
3
4  NOMBRE D EXERCICES PROPOSES : ----
5
6
7  NOMBRE D EXERCICES CORRECTS : ----
8
9
10
11
12
13
14
15
16
17
  ----------------------------------------
```

3-2) Les fenetres de dialogue

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ┌──────────────────────────────────────────────────┐
19 │                                                  │
20 │                                                  │
21 │                                                  │
22 │                                                  │
23 │(ESC = ARRET DE L'AFFICHAGE)                      │
   └──────────────────────────────────────────────────┘
```

D1

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ┌──────────────────────────────────────────────────┐
19 │                                                  │
20 │                                                  │
21 │                                                  │
22 │                                                  │
23 │(ESC = ARRET DE LA CREATION)                      │
   └──────────────────────────────────────────────────┘
```

D2

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ┌──────────────────────────────────────────────────┐
19 │                                                  │
20 │                                                  │
21 │                                                  │
22 │                                                  │
23 │(ESC = ARRET DE LA MODIFICATION)                  │
   └──────────────────────────────────────────────────┘
```

D3

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------

19 DONNEZ UN NOM A LA PREPARATION
20           ---       ----------
21 R:-----------------------------
22
23
   ------------------------------------------------
```

D4

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------

19 QUELLES REPRESENTATIONS UTILISER ?
20 (AU MOINS 1)
21 R:-
22
23
   ------------------------------------------------
```

D5

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------

19 QUELLES REPRESENTATIONS UTILISER ?
20 (AU MOINS 3)
21 R:-
22
23
   ------------------------------------------------
```

D6

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------

19 L'ORIENTATION DU DESSIN PEUT VARIER.
20 POUR QUELLES REPRESENTATIONS ?
21 R:-
22
23
   ------------------------------------------------
```

D7

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------

19 UNE MEME REPRESENTATION PEUT-ELLE ETRE
20 DESSINEE DE DIFFERENTES MANIERES ?
21 R:-
22
23
   ------------------------------------------------
```

D8

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ------------------------------------------------

19 NUMERO DE REPRESENTATION
20
21 R:-
22
23
   ------------------------------------------------
```

D9

```
   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ----------------------------------------------------

19 GENRE D'ASSOCIATIONS
20
21 R:-
22
23
   ----------------------------------------------------

    D10


   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ----------------------------------------------------

19 INTRODUISEZ UN NOMBRE D'EXERCICES
20 COMPRIS ENTRE 00 ET 25.
21
22
23
   ----------------------------------------------------

    D11


   0         1         2         3
   01234567890123456789012345678901234567890123456789
   ----------------------------------------------------

19 REPORT A LA DIZAINE ADMIS ?
20 (O=OUI N=NON)
21 R:-
22
23
   ----------------------------------------------------

    D12
```

```
   0         1         2         3
   01234567890123456789012345678901234567890
   ----------------------------------------

19 VALEUR MAXIMALE DE LA SOMME ?
20 COMPRISE ENTRE 10 ET 100.
21
22
23
   ----------------------------------------
```

D13

```
   0         1         2         3
   01234567890123456789012345678901234567890
   ----------------------------------------

19 SOIT A+B=C. PEUT-ON INTERVERTIR
20 LA LOCALISATION DE A ET DE B ?
21 POUR QUELLES REPRESENTATIONS ?
22 R:-
23
   ----------------------------------------
```

D14

```
   0         1         2         3
   01234567890123456789012345678901234567890
   ----------------------------------------

19 SELECTIONNEZ LES DISTORSIONS ADMISES
20        (AU MOINS 2 PARMI 9 AU TOTAL)
21 R:-
22
23
   ----------------------------------------
```

D15

```
 0         1         2         3
   01234567890123456789012345678901234567890

19 SELECTIONNEZ LA PREPARATION A EXECUTER
20                                 ________
21
22 R:-
23 (ESC = PAS DE SELECTION)
   ----------------------------------------
```

D16

```
   0         1         2         3
   01234567890123456789012345678901234567890

19 SELECTIONNEZ LA PREPARATION A AFFICHER
20                                 ________
21
22 R:-
23 (ESC = PAS DE SELECTION)
   ----------------------------------------
```

D17

```
   0         1         2         3
   01234567890123456789012345678901234567890

19 SELECTIONNEZ LA PREPARATION A MODIFIER
20                                 ________
21
22 R:-
23 (ESC = PAS DE SELECTION)
   ----------------------------------------
```

D18

```
   0         1         2         3
   01234567890123456789012345678901234567890

19 SELECTIONNEZ LA PREPARATION A SUPPRIMER
20                                 _________
21
22 R:-
23 (ESC = PAS DE SELECTION)
   ----------------------------------------
```

D19

annexe 3 : les programmes.

```
(*$S+*)

unit NIVEAU1;intrinsic code 25 data 24;

interface

    uses turtlegraphics;

    type phrase=string[40];

         axex=0..279;

         axey=0..191;

         orient=(n,s,e,o);

         couleur=0..10;

procedure AFFBON (x:axex;
                  y:axey );


procedure EFFBON (x:axex;
                  y:axey);


procedure DESLIG(c:couleur;
                 x1:axex;
                 y1:axey;
                 x2:axex;
                 y2:axey);


procedure AFFCAR(ch:char;
                 x:axex;
                 y:axey);


procedure AFFPHR(x:axex;
                 y:axey;
                 phr:phrase);


procedure DESARC(c:couleur;
                 x1:axex;
                 y1:axey;
                 x2:axex;
                 y2:axey;
                 ri:orient);
```

```
procedure COLREC (c:couleur;
                  x:axex;
                  y:axey;
                  b:axex;
                  h:axey);


procedure DESREC (c:couleur;
                  x:axex;
                  y:axey;
                  b:axex;
                  h:axey);


procedure ELLIPSE (c:couleur;
                   x:axex;
                   y:axey;
                   b:axex;
                   h:axey);




implementation

var tab:array [1..31] of real;

procedure CRAYON(c:couleur);

begin

    case c of

        0 :pencolor(none);
        1 :pencolor(white);
        2 :pencolor(black);
        3 :pencolor(black1);
        4 :pencolor(green);
        5 :pencolor(violet);
        6 :pencolor(white1);
        7 :pencolor(black2);
        8 :pencolor(orange);
        9 :pencolor(blue);
        10:pencolor(white2);

    end;

end;
```

```
procedure AFFBON;

   var des : packed array [1..17,1..9] of boolean;
       i,j :integer;

   begin
       for i:=1 to 17
       do
       begin
         for j:=1 to 9 do begin des[i,j]:= false end;
         end;

       des[17,4] :=true;
       des[17,5] :=true;
       des[17,6] :=true;

       des[16,3] :=true;
       des[16,7] :=true;

       des[15,2] :=true;
       des[15,8] :=true;

       des[14,2] :=true;
       des[14,8] :=true;

       des[13,2] :=true;
       des[13,8] :=true;

       des[12,3] :=true;
       des[12,7] :=true;

       des[11,4] :=true;
       des[11,5] :=true;
       des[11,6] :=true;

       des[10,5] :=true;

       des[9,5] :=true;

       des[8,1]:=true;
       des[8,2]:=true;
       des[8,3]:=true;
       des[8,4]:=true;
       des[8,5]:=true;
       des[8,6]:=true;
       des[8,7]:=true;
       des[8,8]:=true;
       des[8,9]:=true;

       des[7,5]:=true;

       des[6,5]:=true;

       des[5,4]:=true;
       des[5,5]:=true;
       des[5,6]:=true;

       des[4,4]:=true;
       des[4,6]:=true;

       des[3,3]:=true;
```

```
        des[3,7]:=true;

        des[2,3]:=true;
        des[2,7]:=true;

        des[1,2]:=true;
        des[1,8]:=true;
        des[1,1]:=true;
        des[1,9]:=true;

        drawblock(des,2,0,0,9,17,x,y,10);
    end;


procedure EFFBON;

    var des :packed array [1..17,1..9] of boolean;

    begin

        drawblock(des,2,0,0,9,17,x,y,0);

    end;


procedure DESLIG;

begin
    pencolor(none);
    moveto(x1,y1);
    crayon(c);
    moveto(x2,y2);
end;


procedure AFFCAR;

begin
    pencolor(none);
    moveto(x,y);
    wchar(ch);
end;



procedure AFFPHR;

begin

    pencolor(none);
    moveto(x,y);
    chartype(10);
```

```
    wstring(phr);

end;



procedure DESARC;

const t=0.866;

var a,b,r:real;
    cx,x:axex;
    cy,y:axey;
    i,j,k,f1,f2,re:integer;

function FCOS(d:integer) : real;

begin

    case (d div 90) of

    0:fcos:= tab[31-(d div 3)];
    1:fcos:= -tab[31-((180-d) div 3)];
    2:fcos:= -tab[31-((d-180) div 3)];
    3:fcos:= tab[31-((360-d) div 3)];

    end;

end;

function FSIN(d:integer) : real;

begin

    case (d div 90) of

    0:fsin:= tab[d div 3];
    1:fsin:= tab[(180-d) div 3];
    2:fsin:=-tab[(d-180) div 3];
    3:fsin:=-tab[(360-d) div 3];

    end;

end;
begin

    pencolor(none);
    moveto(x1,y1);

    case ri of


        n :begin

                if (x1-x2) <0
                then
                    begin
                        i:=120;
```

```
                    j:=60;
                    f1:=135;
                    f2:=-135;
                end
            else
                begin
                    i:=60;
                    j:=120;
                    f1:=45;
                    f2:=-45;
                end;
            r:=abs(x1-x2)+1;
            a:=(x1+x2)/2;
            b:=y1-(t*r);
            cx:=round(a);
            cy:=round(b+r);


        end;

    s :begin

            if (x1-x2)<0
            then
                begin
                    i:=240;
                    j:=300;
                    f1:=135;
                    f2:=-135;
                end
            else
                begin
                    i:=300;
                    j:=240;
                    f1:=45;
                    f2:=-45;
                end;
            r:=abs(x1-x2)+1;
            a:=(x1+x2)/2;
            b:=y1+(t*r);
            cx:=round(a);
            cy:=round(b-r);
        end;

    e :begin

            if (y1-y2) <0
            then
                begin
                    i:=330;
                    j:=390;
                    f1:=-45;
                    f2:=-135;
                end
            else
                begin
                    i:=390;
                    j:=330;
                    f1:=45;
                    f2:=135;
                end;
            r:=abs(y1-y2)+1;
```

```
            a:=x1-(t*r);
            b:=(y1+y2)/2;
            cx:=round(a+r);
            cy:=round(b);

        end;

    o :begin

            if (y1-y2) <0
            then
                begin
                    i:=210;
                    j:=150;
                    f1:=-45;
                    f2:=-135;
                end
            else
                begin
                    i:=150;
                    j:=210;
                    f1:=45;
                    f2:=135;
                end;
            r:=abs(y1-y2)+1;
            a:=x1+(t*r);
            b:=(y1+y2)/2;
            cx:=round(a-r);
            cy:=round(b);
        end;

end;


crayon(c);

for re:=0 to 20 do

   begin

       if (i-j)<0 then k:=(re*3)+i
                  else k:=i-(re*3);
       x:=round(a+(r*fcos(k)));
       y:=round(b+(r*fsin(k)));
       moveto(x,y);

   end;


pencolor(none);

moveto(cx,cy);

turnto(f1);

crayon(c);

move(5);

pencolor(none);

moveto(cx,cy);
```

```
    turnto(f2);
    crayon(c);
    move(5);
end;

procedure COLREC;

    begin
        viewport(x,x+b,y,y+h);
        case c of
            0 :fillscreen(none);
            1 :fillscreen(white);
            2 :fillscreen(black);
            3 :fillscreen(black1);
            4 :fillscreen(green);
            5 :fillscreen(violet);
            6 :fillscreen(white1);
            7 :fillscreen(black2);
            8 :fillscreen(orange);
            9 :fillscreen(blue);
            10:fillscreen(white2);
        end;
        viewport(0,279,0,191);
    end;

procedure DESREC;

    begin
        pencolor(none);
        moveto(x,y);
        turnto(0);
        crayon(c);
        move(b);
        turn(90);
        move(h);
        turn(90);
        move(b);
        turn(90);
        move(h);
    end;
```

```
procedure ELLIPSE;

var i,e1,e2:integer;


begin


     pencolor(none);
     moveto(x+b,y);
     crayon(c);

     for i:=2 to 31 do
     begin
          e1:=round(b*tab[32-i]);
          e2:=round(h*tab[i]);
          moveto(x+e1,y+e2);
     end;
     for i:=2 to 31 do
     begin
          e1:=round(-b*tab[i]);
          e2:=round(h*tab[32-i]);
          moveto(x+e1,y+e2);
     end;
     for i:=2 to 31 do
     begin
          e1:=round(-b*tab[32-i]);
          e2:=round(-h*tab[i]);
          moveto(x+e1,y+e2);
     end;
     for i:=2 to 31 do
     begin
          e1:=round(b*tab[i]);
          e2:=round(-h*tab[32-i]);
          moveto(x+e1,y+e2);
     end;

end;


begin


tab[1]:=0;
tab[2]:=0.0523;
tab[3]:=0.1045;
tab[4]:=0.1564;
tab[5]:=0.2079;
tab[6]:=0.2588;
tab[7]:=0.3090;
tab[8]:=0.3583;
tab[9]:=0.4067;
tab[10]:=0.4539;
tab[11]:=0.5;
tab[12]:=0.5446;
tab[13]:=0.5877;
tab[14]:=0.6293;
tab[15]:=0.6691;
tab[16]:=0.7071;
```

```
tab[17]:=0.7431;
tab[18]:=0.7771;
tab[19]:=0.8090;
tab[20]:=0.8386;
tab[21]:=0.8660;
tab[22]:=0.8910;
tab[23]:=0.9135;
tab[24]:=0.9335;
tab[25]:=0.9510;
tab[26]:=0.9659;
tab[27]:=0.9781;
tab[28]:=0.9876;
tab[29]:=0.9945;
tab[30]:=0.9986;
tab[31]:=1;

end.
```

```
(*$S+*)

unit NIVEAU2;intrinsic code 26 data 27;

interface

   uses turtlegraphics,transcend,niveau1;

   type message=string[39];

        box=string[3];

        coderep=1..3;

        nocase=1..5;

        noligne=1..4;



   procedure EFFCASE (no:nocase);

   procedure EFFCASE1 ;

   procedure CRECASE1 ;

   procedure CRECASE2 ;

   procedure ANIMBON (noo:nocase;
                      nod:nocase);

   procedure COMMENTAIRE (nolig:noligne;
                          msg:message);

   procedure DIAGRAMME (no:nocase;
                        crp:coderep;
                        sens:orient;
                        casea:box;
                        caseb:box;
                        casec:box;
                        cola:couleur;
                        colb:couleur;
                        colc:couleur);

   procedure ARBRE (no:nocase;
                    sens:orient;
                    casea:box;
                    caseb:box;
                    casec:box;
                    cola:couleur;
                    colb:couleur;
                    colc:couleur);

   procedure GRAPHE (no:nocase;
                     cg:coderep;
                     sens:orient;
                     casea:box;
                     caseb:box;
                     casec:box;
                     cola:couleur;
                     colb:couleur;
```

```
                        colc:couleur);

   procedure LINEAIRE (no:nocase;
                       sens:orient;
                       casea:box;
                       caseb:box;
                       casec:box;
                       cola:couleur;
                       colb:couleur;
                       colc:couleur);

   procedure ARTPOS (no:nocase;
                     sens:orient;
                     casea:box;
                     caseb:box;
                     casec:box;
                     cola:couleur;
                     colb:couleur;
                     colc:couleur);

   procedure ARTNEG (no:nocase;
                     sens:orient;
                     casea:box;
                     caseb:box;
                     casec:box;
                     cola:couleur;
                     colb:couleur;
                     colc:couleur);


implementation


const b=23;
      h=12;

var dx:axex;
    dy:axey;

procedure DEPLACE (no:nocase);

 begin

     case no of

     1:begin
           dx:=2;
           dy:=118;
       end;
     2:begin
           dx:=140;
           dy:=118;
       end;
     3:begin
           dx:=2;
           dy:=46;
       end;
     4:begin
           dx:=140;
           dy:=46;
       end;
```

```
     5:begin
           dx:=2;
           dy:=2;
       end;

     end;
end;



procedure EFFCASE;

begin
    deplace(no);
    if no=5 then colrec(2,dx+1,dy+1,273,39)
            else colrec(2,dx+1,dy+1,135,69);
end;


procedure EFFCASE1;


begin
    deplace(1);
    colrec(2,dx+2,dy+3,132,66);
end;


procedure CRECASE2;

begin
    colrec(1,0,0,279,191);
    effcase(1);
    effcase(2);
    effcase(3);
    effcase(4);
    effcase(5);
end;

procedure CRECASE1;

begin
    colrec(1,0,0,279,191);
    effcase1;
    effcase(2);
    effcase(3);
    effcase(4);
    effcase(5);
end;



procedure COMMENTAIRE;

begin
    deplace(5);

    case nolig of
    1:affphr(dx+2,dy+32,msg);
    2:affphr(dx+2,dy+22,msg);
    3:affphr(dx+2,dy+12,msg);
```

```
    4:affphr(dx+2,dy+2,msg);
    end;

end;



procedure ANIMBON;

begin
    deplace(noo);
    effbon(dx+125,dy+3);
    deplace(nod);
    affbon(dx+125,dy+3);
end;



procedure DIAGRAMME ;


var
   ce1x,ce2x,ce3x,bpx,hpx,rgx,drgx,rdx,drdx,rox,drox,argx,ardx,arox
   ,bgx:axex;

   ce1y,ce2y,ce3y,bpy,hpy,rgy,drgy,rdy,drdy,roy,droy,argy,ardy,aroy
   ,hgy:axey;

   rd,rg,ro:box;

   colg,cold,colo:couleur;



    procedure SOS1;

    begin


    if (sens=n) or (sens=s)

    then
        begin

        ce1x:=45;
        ce1y:=35;

        ce2x:=90;
        ce2y:=35;

        ce3x:=68;
        ce3y:=35;

        bpx:=20;
        hpy:=10;

        bgx:=50;
```

```
hgy:=15;

if sens=n

then
    begin

        rgx:=34;
        rgy:=5;
        drgx:=45;
        drgy:=17;

        rdx:=79;
        rdy:=5;
        drdx:=90;
        drdy:=17;

        case crp of

            1:begin
                rox:=57;
                roy:=53;
                drox:=68;
                droy:=52;

                argx:=45;
                argy:=24;

                ardx:=90;
                ardy:=24;

                arox:=68;
                aroy:=51;

                rg:=casea;
                rd:=caseb;
                ro:=casec;

                colg:=cola;
                cold:=colb;
                colo:=colc;

              end;

            2:begin
                rox:=34;
                roy:=53;
                drox:=45;
                droy:=52;

                argx:=45;
                argy:=21;

                ardx:=90;
                ardy:=25;

                arox:=45;
                aroy:=46;

                rg:=casec;
                rd:=caseb;
                ro:=casea;
```

```
                    colg:=colc;
                    cold:=colb;
                    colo:=cola;

                end;

            3:begin
                    rox:=79;
                    roy:=53;
                    drox:=90;
                    droy:=52;

                    argx:=45;
                    argy:=25;

                    ardx:=90;
                    ardy:=21;

                    arox:=90;
                    aroy:=45;

                    rg:=casea;
                    rd:=casec;
                    ro:=caseb;

                    colg:=cola;
                    cold:=colc;
                    colo:=colb;

                end;

        end;

    end

else

    begin
        rgx:=34;
        rgy:=53;
        drgx:=45;
        drgy:=52;

        rdx:=79;
        rdy:=53;
        drdx:=90;
        drdy:=52;

        case crp of

            1:begin

                    rox:=57;
                    roy:=5;
                    drox:=68;
                    droy:=17;

                    argx:=45;
                    argy:=46;

                    ardx:=90;
```

```
            ardy:=46;

            arox:=68;
            aroy:=19;

            rg:=casea;
            rd:=caseb;
            ro:=casec;

            colg:=cola;
            cold:=colb;
            colo:=colc;

        end;

    2:begin

            rox:=34;
            roy:=5;
            drox:=45;
            droy:=17;

            argx:=45;
            argy:=48;

            ardx:=90;
            ardy:=46;

            arox:=45;
            aroy:=24;

            rg:=casec;
            rd:=caseb;
            ro:=casea;

            colg:=colc;
            cold:=colb;
            colo:=cola;

        end;

    3:begin

            rox:=79;
            roy:=5;
            drox:=90;
            droy:=17;

            argx:=45;
            argy:=46;

            ardx:=90;
            ardy:=48;

            arox:=90;
            aroy:=24;

            rg:=casea;
            rd:=casec;
            ro:=caseb;
```

```
                        colg:=cola;
                        cold:=colc;
                        colo:=colb;

                    end;

                end;

            end;

        end;

end;



procedure SOS2;

begin

if (sens=e) or (sens=o)

then

        begin

            ce1x:=68;
            ce1y:=50;

            ce2x:=68;
            ce2y:=20;

            ce3x:=68;
            ce3y:=35;

            bpx:=10;
            hpy:=13;

            bgx:=20;
            hgy:=30;

            if sens=e

            then
                begin

                    rgx:=17;
                    rgy:=14;
                    drgx:=41;
                    drgy:=20;

                    rdx:=17;
                    rdy:=44;
                    drdx:=41;
                    drdy:=50;

                    case crp of

                        1:begin
```

```
            rox:=98;
            roy:=29;
            drox:=97;
            droy:=35;

            argx:=57;
            argy:=20;

            ardx:=57;
            ardy:=50;

            arox:=89;
            aroy:=35;

            rg:=caseb;
            rd:=casea;
            ro:=casec;

            colg:=colb;
            cold:=cola;
            colo:=colc;

        end;

    2:begin
            rox:=98;
            roy:=44;
            drox:=97;
            droy:=50;

            argx:=57;
            argy:=20;

            ardx:=50;
            ardy:=50;

            arox:=79;
            aroy:=50;

            rg:=caseb;
            rd:=casec;
            ro:=casea;

            colg:=colb;
            cold:=colc;
            colo:=cola;

        end;

    3:begin
            rox:=98;
            roy:=14;
            drox:=97;
            droy:=20;

            argx:=50;
            argy:=20;

            ardx:=57;
            ardy:=50;

            arox:=79;
```

```
                    aroy:=20;

                    rg:=casec;
                    rd:=casea;
                    ro:=caseb;

                    colg:=colc;
                    cold:=cola;
                    colo:=colb;

                end;

            end;

        end

    else

        begin

            rgx:=98;
            rgy:=14;
            drgx:=97;
            drgy:=20;

            rdx:=98;
            rdy:=44;
            drdx:=97;
            drdy:=50;

            case crp of

                1:begin
                    rox:=18;
                    roy:=29;
                    drox:=41;
                    droy:=35;

                    argx:=79;
                    argy:=20;

                    ardx:=79;
                    ardy:=50;

                    arox:=48;
                    aroy:=35;

                    rg:=caseb;
                    rd:=casea;
                    ro:=casec;

                    colg:=colb;
                    cold:=cola;
                    colo:=colc;

                  end;

                2:begin
                    rox:=18;
                    roy:=44;
                    drox:=41;
                    droy:=50;
```

```
                    argx:=79;
                    argy:=20;

                    ardx:=85;
                    ardy:=50;

                    arox:=57;
                    aroy:=50;

                    rg:=caseb;
                    rd:=casec;
                    ro:=casea;

                    colg:=colb;
                    cold:=colc;
                    colo:=cola;

                end;

            3:begin
                    rox:=18;
                    roy:=14;
                    drox:=41;
                    droy:=20;

                    argx:=85;
                    argy:=20;

                    ardx:=79;
                    ardy:=50;

                    arox:=57;
                    aroy:=20;

                    rg:=casec;
                    rd:=casea;
                    ro:=caseb;

                    colg:=colc;
                    cold:=cola;
                    colo:=colb;

                end;

            end;

        end;

    end;

end;


procedure SOS3;

begin

    ellipse(cola,ce1x+dx,ce1y+dy,bpx,hpy);
    ellipse(colb,ce2x+dx,ce2y+dy,bpx,hpy);
```

```
      ellipse(colc,ce3x+dx,ce3y+dy,bgx,hgy);

      desrec(colg,rgx+dx,rgy+dy,b,h);
      desrec(cold,rdx+dx,rdy+dy,b,h);
      desrec(colo,rox+dx,roy+dy,b,h);

      deslig(cold,drdx+dx,drdy+dy,ardx+dx,ardy+dy);
      deslig(colg,drgx+dx,drgy+dy,argx+dx,argy+dy);
      deslig(colo,drox+dx,droy+dy,arox+dx,aroy+dy);

      affphr(rgx+dx+2,rgy+dy+2,rg);
      affphr(rdx+dx+2,rdy+dy+2,rd);
      affphr(rox+dx+2,roy+dy+2,ro);


 end;


 begin

      deplace(no);
      sos1;
      sos2;
      sos3;

 end;



procedure ARBRE ;

var
     cx,mcx,ax,max,bx,mbx:axex;

     cy,mcy,ay,may,by,mby:axey;


begin

     case sens  of

          n:begin

                cx:=57;
                cy:=55;
                mcx:=68;
                mcy:=54;

                ax:=36;
                ay:=5;
                max:=47;
                may:=17;

                bx:=78;
                by:=5;
                mbx:=89;
                mby:=17;

            end;
```

```
      s:begin

            cx:=57;
            cy:=5;
            mcx:=68;
            mcy:=17;

            ax:=36;
            ay:=55;
            max:=46;
            may:=54;

            bx:=78;
            by:=55;
            mbx:=89;
            mby:=54;

        end;

      o:begin

            cx:=27;
            cy:=30;
            mcx:=50;
            mcy:=35;

            ax:=88;
            ay:=50;
            max:=87;
            may:=55;

            bx:=88;
            by:=10;
            mbx:=87;
            mby:=15;

        end;

      e:begin

            cx:=88;
            cy:=30;
            mcx:=87;
            mcy:=35;

            ax:=27;
            ay:=50;
            max:=50;
            may:=55;

            bx:=27;
            by:=10;
            mbx:=50;
            mby:=15;

        end;

  end;

  deplace(no);
```

```
      desrec(colc,cx+dx,cy+dy,b,h);

      deslig(cola,mcx+dx,mcy+dy,max+dx,may+dy);

      deslig(colb,mcx+dx,mcy+dy,mbx+dx,mby+dy);

      desrec(cola,ax+dx,ay+dy,b,h);

      desrec(colb,bx+dx,by+dy,b,h);

      affphr(cx+dx+2,cy+dy+2,casec);

      affphr(ax+dx+2,ay+dy+2,casea);

      affphr(bx+dx+2,by+dy+2,caseb);

 end;


procedure GRAPHE ;

var

   colg,cold:couleur;

   rgx,rdx,dfhx,afhx,rhx,dfbx,afbx,rbx,ofhx:axex;

   rgy,rdy,dfhy,afhy,rhy,dfby,afby,rby,ofhy:axey;

   rg,rd:box;

   carh,carb:char;

   ofh,ofb:orient;


begin

   if (sens=n) or (sens=s)

   then

   begin

      rgx:=15;
      rgy:=30;

      rdx:=100;
      rdy:=30;

      rhx:=61;
      rhy:=55;

      rbx:=61;
      rby:=5;

      ofh:=n;
```

```
ofb:=s;

case cg of

1:begin

     dfhx:=39;
     dfhy:=36;
     dfbx:=99;
     dfby:=35;
     afhx:=99;
     afhy:=36;
     afbx:=39;
     afby:=35;

     if (sens=n)

     then

     begin

        carh:='+';
        carb:='-';
        rg:=casea;
        rd:=casec;
        colg:=cola;
        cold:=colc;

     end

     else

     begin

        carh:='-';
        carb:='+';
        rg:=casec;
        rd:=casea;
        colg:=colc;
        cold:=cola;

     end;


  end;


2:begin

     dfhx:=99;
     dfhy:=36;
     dfbx:=39;
     dfby:=35;
     afhx:=39;
     afhy:=36;
     afbx:=99;
     afby:=35;

     if (sens=n)

     then
```

```
          begin

             carh:='+';
             carb:='-';
             rg:=casec;
             rd:=casea;
             colg:=colc;
             cold:=cola;

          end

          else

          begin

             carh:='-';
             carb:='+';
             rg:=casea;
             rd:=casec;
             colg:=cola;
             cold:=colc;

          end;

          end;


        end;

end;


 if (sens=e) or (sens=o)

 then

 begin

    rgx:=56;
    rgy:=5;

    rdx:=56;
    rdy:=55;

    rhx:=29;
    rhy:=30;

    rbx:=91;
    rby:=30;

    ofh:=o;
    ofb:=e;

    case cg of

    1:begin

         dfhx:=68;
         dfhy:=18;
         dfbx:=68;
         dfby:=54;
```

```
            afhx:=68;
            afhy:=54;
            afbx:=68;
            afby:=18;

        if (sens=e)

        then

        begin

            carh:='-';
            carb:='+';
            rg:=casec;
            rd:=casea;
            colg:=colc;
            cold:=cola;

        end

        else

        begin

            carh:='+';
            carb:='-';
            rg:=casea;
            rd:=casec;
            colg:=cola;
            cold:=colc;

        end;

    end;


2:begin

        dfhx:=68;
        dfhy:=54;
        dfbx:=68;
        dfby:=18;
        afhx:=68;
        afhy:=18;
        afbx:=68;
        afby:=54;


        if (sens=e)

        then

        begin

            carh:='-';
            carb:='+';
            rg:=casea;
            rd:=casec;
            colg:=cola;
            cold:=colc;

        end
```

```
        else

        begin

           carh:='+';
           carb:='-';
           rg:=casec;
           rd:=casea;
           colg:=colc;
           cold:=cola;

        end;

      end;

   end;

   end;


   deplace(no);

   desrec(colg,rgx+dx,rgy+dy,b,h);

   affphr(rgx+dx+2,rgy+dy+2,rg);

   desrec(cold,rdx+dx,rdy+dy,b,h);

   affphr(rdx+dx+2,rdy+dy+2,rd);

   desarc(1,dfhx+dx,dfhy+dy,afhx+dx,afhy+dy,ofh);

   desrec(colb,rhx+dx,rhy+dy,b,h);

   affcar(carh,rhx+dx-8,rhy+dy+2);

   affphr(rhx+dx+2,rhy+dy+2,caseb);

   desarc(1,dfbx+dx,dfby+dy,afbx+dx,afby+dy,ofb);

   desrec(colb,rbx+dx,rby+dy,b,h);

   affcar(carb,rbx+dx-8,rby+dy+2);

   affphr(rbx+dx+2,rby+dy+2,caseb);

end;


procedure LINEAIRE ;

type strg1=string[1];

var
    debuta,debutb,debutc,longa,longb,longc,dca,dcb,dcc,base:axex;

    y1,y2:axey;
```

```
    ;
    pcasea,pcaseb,pcasec:box;

    a,b,c,val1,val2,val3,val4,val5:integer;


procedure RESSERRER(cas:box;
                    var pcas:box);

begin
    pcas:=cas;

    while (pos(' ',pcas) <> 0 )

    do delete(pcas,pos(' ',pcas),1);

end;


function VALEUR (chiffre:strg1) :integer;

begin

    valeur:=pos(chiffre,'0123456789')-1;

end;




function CALCUL (pcas:box):integer;

var long,unite,dizaine,centaine:integer;


begin

    unite:=0;
    dizaine:=0;
    centaine:=0;

    long:=length(pcas);

    if long=1
    then unite:=valeur(pcas);

    if long=2
    then begin
              unite:=valeur(copy(pcas,long,1));
              dizaine:=valeur(copy(pcas,long-1,1));
         end;

    if long=3
    then begin
              unite:=valeur(copy(pcas,long,1));
              dizaine:=valeur(copy(pcas,long-1,1));
              centaine:=valeur(copy(pcas,long-2,1));
         end;

    calcul:=unite+(dizaine*10)+(centaine*100);
```

```
end;



procedure CALCASE (pcase1:box;
                   pcase2:box;
                   pcase3:box;
                   var a:integer;
                   var b:integer;
                   var c:integer);


begin
    if pcase1='?'
    then
       begin
           b:=calcul(pcase2);
           c:=calcul(pcase3);
           a:=c-b;
       end;

    if pcase2='?'
    then
       begin
           a:=calcul(pcase1);
           c:=calcul(pcase3);
           b:=c-a;
       end;


    if pcase3 = '?'

    then begin

             a:=calcul(pcase1);
             b:=calcul(pcase2);
             c:=a+b;

         end;


end;




function MAX (x:integer;y:integer):integer;

begin
    if x>=y then max:=x else max:=y;
end;




begin
```

```
    case sens of

    n:begin
            y1:=23;
            y2:=37;
        end;

    s:begin
            y1:=37;
            y2:=23;
        end;

    end;


    resserrer(casea,pcasea);
    resserrer(caseb,pcaseb);
    resserrer(casec,pcasec);
    calcase(pcasea,pcaseb,pcasec,a,b,c);

    base:= 125 div c;
    if base > 7 then base:=7;

    val1:=length(pcasea)*7;
    val2:=a*base;
    val3:=length(pcaseb)*7;
    val4:=b*base;
    val5:=length(pcasec)*7;

    longa:=max(val1,val2)+2;
    longb:=max(val3,val4)+2;
    longc:=longa+longb+2;

    debutc:=68-(longc div 2);
    debuta:=debutc;
    debutb:=debuta+longa+2;


    dca:=debuta+((longa-val1) div 2);
    dcb:=debutb+((longb-val3) div 2);
    dcc:=debutc+((longc-val5) div 2);


    deplace(no);

    desrec(cola,debuta+dx,y2+dy,longa,h);

    affphr(dca+dx,y2+dy+2,pcasea);

    desrec(colb,debutb+dx,y2+dy,longb,h);

    affphr(dcb+dx,y2+dy+2,pcaseb);

    desrec(colc,debutc+dx,y1+dy,longc,h);

    affphr(dcc+dx,y1+dy+2,pcasec);

end;
```

```
procedure ARTPOS ;

var
   r1x,r2x,r3x,car1x,car2x:axex;

   r1y,r2y,r3y,car1y,car2y:axey;

   r1,r2,r3:box;

   car1,car2:char;

   col1,col2,col3:couleur;


begin

    case sens of

    n:begin
          car1:='+';
          car2:='=';
          r1:=casea;
          r2:=caseb;
          r3:=casec;
          col1:=cola;
          col2:=colb;
          col3:=colc;
      end;

    s:begin
          car1:='=';
          car2:='+';
          r1:=casec;
          r2:=casea;
          r3:=caseb;
          col1:=colc;
          col2:=cola;
          col3:=colb;
      end;

    end;

    r1x:=23;
    r1y:=30;
    r2x:=57;
    r2y:=30;
    r3x:=91;
    r3y:=30;
    car1x:=49;
    car1y:=31;
    car2x:=83;
    car2y:=31;



    deplace(no);

    desrec(col1,r1x+dx,r1y+dy,b,h);
```

```
    affphr(r1x+dx+2,r1y+dy+2,r1);
    affcar(car1,car1x+dx,car1y+dy);
    desrec(col2,r2x+dx,r2y+dy,b,h);
    affphr(r2x+dx+2,r2y+dy+2,r2);
    affcar(car2,car2x+dx,car2y+dy);
    desrec(col3,r3x+dx,r3y+dy,b,h);
    affphr(r3x+dx+2,r3y+dy+2,r3);
end;



procedure ARTNEG;

var
   r1x,r2x,r3x,car1x,car2x:axex;
   r1y,r2y,r3y,car1y,car2y:axey;
   r1,r2,r3:box;
   car1,car2:char;
   col1,col2,col3:couleur;


begin
    case sens of
    n:begin
          car1:='-';
          car2:='=';
          r1:=casec;
          r2:=caseb;
          r3:=casea;
          col1:=colc;
          col2:=colb;
          col3:=cola;
      end;
    s:begin
          car1:='=';
          car2:='-';
          r1:=casea;
          r2:=casec;
          r3:=caseb;
          col1:=cola;
          col2:=colc;
          col3:=colb;
      end;
    end;
```

```
    r1x:=23;
    r1y:=30;
    r2x:=57;
    r2y:=30;
    r3x:=91;
    r3y:=30;
    car1x:=49;
    car1y:=31;
    car2x:=83;
    car2y:=31;


    deplace(no);

    desrec(col1,r1x+dx,r1y+dy,b,h);

    affphr(r1x+dx+2,r1y+dy+2,r1);

    affcar(car1,car1x+dx,car1y+dy);

    desrec(col2,r2x+dx,r2y+dy,b,h);

    affphr(r2x+dx+2,r2y+dy+2,r2);

    affcar(car2,car2x+dx,car2y+dy);

    desrec(col3,r3x+dx,r3y+dy,b,h);

    affphr(r3x+dx+2,r3y+dy+2,r3);

end;


begin
end.
```

```
(*$S+*)

unit FICHIER:intrinsic code 18 data 19;

interface

uses applestuff,ecran;

procedure OUVRE(f:nofich);
procedure FERME(f:nofich);
procedure LECTURE(f:nofich;
                  i:indfich);
procedure ECRITURE(f:nofich;
                   i:indfich);
procedure EXAFICH(f:nofich;
                  var i:indfich);


implementation

var ch:char;
    res:integer;

procedure OUVRE;
begin
(*$I-*)
    case f of
    1:reset(f1,'#5:PRE/1/SEL.DATA');
    2:reset(f2,'#5:PRE/2/SEL.DATA');
    3:reset(f3,'#5:PRE/3/ASS.DATA');
    end;
    res:=ioresult;
    if res<>0 then
    begin
        effecr;
        gotoxy(4,10);
        write('ERREUR D ENTREE/SORTIE NO ',res);
        gotoxy(4,12);
        write('SUR LE FICHIER NO ',f);
        gotoxy(4,14);
        write('LORS D UNE OPERATION D OUVERTURE');
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
        while ord(ch)<>32 do read(keyboard,ch);
        exit(program);
    END;
(*$I+*)
end;

procedure FERME;

begin
(*$I-*)
    case f of
    1:close(f1);
    2:close(f2);
    3:close(f3);
    end;
    res:=ioresult;
    if res<>0 then
    begin
```

```
        effecr;
        gotoxy(4,10);
        write('ERREUR D ENTREE/SORTIE NO ',res);
        gotoxy(4,12);
        write('SUR LE FICHIER NO ',f);
        gotoxy(4,14);
        write('LORS D UNE OPERATION DE FERMETURE ');
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
        while ord(ch)<>32 do read(keyboard,ch);
        exit(program);
    END;
(*$I+*)
end;

procedure LECTURE;
begin
(*$I-*)
    case f of
    1:begin
      seek(f1,i);
      get(f1);
      end;
    2:begin
      seek(f2,i);
      get(f2);
      end;
    3:begin
      seek(f3,i);
      get(f3);
      end;
    end;
    res:=ioresult;
    if res<>0 then
    begin
        effecr;
        gotoxy(4,10);
        write('ERREUR D ENTREE/SORTIE NO ',res);
        gotoxy(4,12);
        write('SUR LE FICHIER NO ',f);
        gotoxy(4,14);
        write('LORS D UNE OPERATION DE LECTURE ');
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
        while ord(ch)<>32 do read(keyboard,ch);
        exit(program);
    END;
(*$I+*)
end;


procedure ECRITURE;
begin
(*$I-*)
    case f of
    1:begin
      seek(f1,i);
      put(f1);
      end;
    2:begin
```

```
      seek(f2,i);
      put(f2);
      end;
    3:begin
      seek(f3,i);
      put(f3);
      end;
    end;
    res:=ioresult;
    if res<>0 then
    begin
        effecr;
        gotoxy(4,10);
        write('ERREUR D ENTREE/SORTIE NO ',res);
        gotoxy(4,12);
        write('SUR LE FICHIER NO ',f);
        gotoxy(4,14);
        write('LORS D UNE OPERATION D ECRITURE ');
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
        while ord(ch)<>32 do read(keyboard,ch);
        exit(program);
    END;
(*$I+*)
end;

procedure EXAFICH;
begin
    i:=0;
    ouvre(f);
    lecture(f,i);
    case f of
    1:begin
      while( f1^.nomprepa<>'                              ')
            and (i<9) do
      begin
          i:=i+1;
          lecture(f,i);
      end;
      if f1^.nomprepa='                                   '
      then i:=i-1;
      end;
    2:begin
      while( f2^.nomprepa<>'                              ')
            and (i<9) do
      begin
          i:=i+1;
          lecture(f,i);
      end;
      if f2^.nomprepa='                                   '
      then i:=i-1;
      end;
    3:begin
      while( f3^.nomprepa<>'                              ')
            and (i<9) do
      begin
          i:=i+1;
          lecture(f,i);
      end;
      if f3^.nomprepa='                                   '
      then i:=i-1;
```

```
      end;
    end;

    ferme(f);
end;

begin
end.
```

```
(*$S+*)

unit ECRAN:intrinsic code 16 data 17;

interface

const maxindfich=9;

type

    numex=0..25;

    prepasel    =record

                 nomprepa:packed array[1..25] of char;
                 reputil:packed array[1..6] of boolean;
                 vardess:packed array[1..2] of boolean;
                 varspat:packed array[1..6] of boolean;
                 interve:packed array[1..6] of boolean;

                 nbrex:packed array[1..6,1..6] of numex;

                 valmax:10..100;

                 passdix:boolean;

                 tabsit1:packed array[1..9] of boolean;
                 tabsit2:packed array[1..9] of boolean;

                 end;

    prepass    =record

                 nomprepa:packed array[1..25] of char;
                 reputil:packed array[1..6] of boolean;
                 vardess:packed array[1..2] of boolean;
                 varspat:packed array[1..6] of boolean;
                 interve:packed array[1..6] of boolean;

                 nbrex:packed array[1..5] of numex;

                 valmax:10..100;

                 passdix:boolean;

                 tabsit1:packed array[1..9] of boolean;
                 tabsit2:packed array[1..9] of boolean;

                 end;

    fichsel=file of prepasel;
    fichass=file of prepass;
    nofich=1..3;
    indfich=0..maxindfich;
    coorx=0..39;
    coory=0..23;

var
   f1,f2:fichsel;
   f3:fichass;
```

```
procedure EFFECR;
procedure EFFECR1;
procedure EFFECR2;
procedure EFFECR3;
procedure LIGNE;


implementation

var i:coory;

procedure EFFECR;
begin
    for i:=0 to 23 do
    begin
        gotoxy(0,i);
        write('                                                  ');
    end;
end;

procedure EFFECR1;
begin
    for i:=2 to 23 do
    begin
        gotoxy(0,i);
        write('                                                  ');
    end;
end;

procedure EFFECR2;
begin
    for i:=19 to 23 do
    begin
        gotoxy(0,i);
        write('                                                  ');
    end;
end;

procedure EFFECR3;
begin
    for i:=5 to 17 do
    begin
        gotoxy(0,i);
        write('                                                  ');
    end;
end;


procedure LIGNE;
begin
    gotoxy(0,18);
    write('==========================================');
end;




begin
```

```
end.
```

```
segment procedure PREPARE;

type tmode=1..6;
     tsmode=(a,m,s);
     msg=1..8;

var ok:boolean;
    ch:char;
    mo:tmode;
    f:nofich;
    bs,i:indfich;
    fini:boolean;
    q:integer;

procedure SELMODE(var m:tmode);

begin
    gotoxy(10,1);
    write('MENU PREPARATION');
    gotoxy(10,2);
    write('================');
    gotoxy(3,4);
    write('1-LISTER LES PREPARATIONS EXISTANTES');
    gotoxy(3,6);
    write('2-AFFICHER UNE PREPARATION');
    gotoxy(3,8);
    write('3-CREER UNE PREPARATION');
    gotoxy(3,10);
    write('4-MODIFIER UNE PREPARATION');
    gotoxy(3,12);
    write('5-SUPPRIMER UNE PREPARATION');
    gotoxy(3,14);
    write('6-FIN DE PREPARATION');
    gotoxy(3,17);
    write('VOTRE CHOIX :-');
    gotoxy(3,18);
    write('------------');
    gotoxy(16,17);

    ok:=false;
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while not ok do
    begin
        while (ch<>'1')and
              (ch<>'2')and
              (ch<>'3')and
              (ch<>'4')and
              (ch<>'5')and
              (ch<>'6') do
        begin
            gotoxy(16,17);
            write('-');
            gotoxy(16,17);
            read(keyboard,ch);
        end;
        gotoxy(16,17);
        write(ch);
        case ch of
        '1':gotoxy(3,4);
```

```
        '2':gotoxy(3,6);
        '3':gotoxy(3,8);
        '4':gotoxy(3,10);
        '5':gotoxy(3,12);
        '6':gotoxy(3,14);
        end;
        m:=ord(ch)-48;
        read(keyboard,ch);
        if ch=chr(32) then ok:=true;
    end;
end;

procedure SELEX(mode:tmode;
                var f:nofich);


procedure TITRE(mode:tmode);
begin
case mode of
1:begin
      gotoxy(5,0);
      write('LISTER LES PREPARATIONS EXISTANTES');
      gotoxy(5,1);
      write('====================================');
  end;
2:begin
      gotoxy(9,0);
      write('AFFICHER UNE PREPARATION');
      gotoxy(9,1);
      write('=========================');
  end;
3:begin
      gotoxy(11,0);
      write('CREER UNE PREPARATION');
      gotoxy(11,1);
      write('======================');
  end;
4:begin
      gotoxy(9,0);
      write('MODIFIER UNE PREPARATION');
      gotoxy(9,1);
      write('=========================');
  end;
5:begin
      gotoxy(8,0);
      write('SUPPRIMER UNE PREPARATION');
      gotoxy(8,1);
      write('==========================');
  end;
end;
end;

begin
    titre(mode);
    gotoxy(0,3);
    write('EXERCICES DE RECONNAISSANCE :');
    gotoxy(0,6);
    write('1-SELECTION PARMI 3 SITUATIONS EXPRIMEES');
    gotoxy(2,7);
    write('AVEC LA MEME REPRESENTATION');
```

```
    gotoxy(0,9);
    write('2-SELECTION PARMI 3 SITUATIONS EXPRIMEES');
    gotoxy(2,10);
    write('AVEC DES REPRESENTATIONS DIFFERENTES');
    gotoxy(0,12);
    write('3-ASSOCIATION DES SITUATIONS SEMBLABLES');
    gotoxy(0,16);
    write('VOTRE CHOIX :-');
    gotoxy(0,17);
    write('------------');
    gotoxy(13,16);
    ok:=false;
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while not ok do
    begin
        while (ch<>'1')and
              (ch<>'2')and
              (ch<>'3') do
        begin
            gotoxy(13,16);
            write('-');
            gotoxy(13,16);
            read(keyboard,ch);
        end;
        gotoxy(13,16);
        write(ch);
        case ch of
        '1':gotoxy(0,6);
        '2':gotoxy(0,9);
        '3':gotoxy(0,12);
        end;
        f:=ord(ch)-48;
        read(keyboard,ch);
        if ch=chr(32) then ok:=true;
    end;
end;


procedure SELPREPA(f:nofich;
                   smode:tsmode;
                   bs:indfich;
                   var fin:boolean;
                   var sel:indfich );

var bi,i:indfich;
    verbe,sligne:string;
    x:coorx;
    y:coory;

procedure AFFEXER;
begin
    gotoxy(0,3);
    write('EXERCICES DE RECONNAISSANCE :');
    gotoxy(0,4);
    case f of
    1:begin
          write('SELECTION PARMI 3 SITUATIONS EXPRIMEES');
          gotoxy(0,5);
          write('AVEC LA MEME REPRESENTATION');
      end;
```

```
    2:begin
          write('SELECTION PARMI 3 SITUATIONS EXPRIMEES');
          gotoxy(0,5);
          write('AVEC DES REPRESENTATIONS DIFFERENTES');
      end;
    3:write('ASSOCIATIONS DES SITUATIONS');
    end;

end;

begin
case smode of
a:begin
      gotoxy(9,0);
      write('AFFICHER UNE PREPARATION');
      gotoxy(9,1);
      write('==========================');
      bi:=0;
      verbe:='AFFICHER';
      sligne:='--------';
  end;
m:begin
      gotoxy(9,0);
      write('MODIFIER UNE PREPARATION');
      gotoxy(9,1);
      write('==========================');
      bi:=1;
      verbe:='MODIFIER';
      sligne:='--------';
  end;
s:begin
      gotoxy(8,0);
      write('SUPPRIMER UNE PREPARATION');
      gotoxy(8,1);
      write('==========================');
      bi:=1;
      verbe:='SUPPRIMER';
      sligne:='----------';
  end;
end;


affexer;

    ouvre(f);
    x:=0;
    y:=7;
    for i:=bi to maxindfich do
    begin
        gotoxy(x,y);
        lecture(f,i);
        write(i,'-');
        case f of
        1:write(f1^.nomprepa);
        2:write(f2^.nomprepa);
        3:write(f3^.nomprepa);
        end;
        y:=y+1;
    end;
    y:=7;
    ferme(f);
    ligne;
```

```
    gotoxy(0,19);
    write('SELECTIONNEZ LA PREPARATION A ',verbe);
    gotoxy(30,20);
    write(signe);

    gotoxy(0,22);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = PAS DE SELECTION)');
    gotoxy(2,22);
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    ok:=false;
    fin:=false;
    while not ok do
    begin
        while (ord(ch)-48 < bi) or
              (ord(ch)-48 > bs) do
        begin
            if ord(ch)=27 then begin
                               fin:=true;
                               exit(selprepa);
                               end;

            gotoxy(2,22);
            write('-');
            gotoxy(2,22);
            read(keyboard,ch);
        end;
        gotoxy(2,22);
        write(ch);
        sel:=ord(ch)-48;
        gotoxy(0,y+(ord(ch)-48-bi));
        read(keyboard,ch);
        if ch=chr(32) then ok:=true
        else if ord(ch)=27 then begin
                                fin:=true;
                                exit(selprepa);
                                end;

    end;

end;


procedure LISTEPREPA(f:nofich);
var i:indfich;

begin
    gotoxy(4,0);
    write('LISTE DES PREPARATIONS EXISTANTES');
    gotoxy(4,1);
    write('================================');

    gotoxy(0,3);
    write('EXERCICES DE RECONNAISSANCE :');
    gotoxy(0,4);
    case f of
    1:begin
        write('SELECTION PARMI 3 SITUATIONS EXPRIMEES');
```

```
          gotoxy(0,5);
          write('AVEC LA MEME REPRESENTATION');
      end;
    2:begin
          write('SELECTION PARMI 3 SITUATIONS EXPRIMEES');
          gotoxy(0,5);
          write('AVEC DES REPRESENTATIONS DIFFERENTES');
      end;
    3:write('ASSOCIATIONS DES SITUATIONS');
    end;

    ouvre(f);
    for i:=0 to maxindfich do
    begin
        gotoxy(0,7+i);
        lecture(f,i);
        write(i,'-');
        case f of
        1:write(f1^.nomprepa);
        2:write(f2^.nomprepa);
        3:write(f3^.nomprepa);
        end;
    end;
    ferme(f);
    ligne;

    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ch<>chr(32) do read(keyboard,ch);
end;



procedure MESSAGE(no:msg);
var s1,s2:string[40];
begin
    effecr;
    case no of
    1:begin
      s1:='CREATION IMPOSSIBLE :';
      s2:='10 PREPARATIONS ONT DEJA ETE CREES';
      end;
    2:begin
      s1:='CREATION INTERROMPUE :';
      s2:='ACUNE INFORMATION N EST ENREGISTREE';
      end;
    3:begin
      s1:='CREATION TERMINEE :';
      s2:='LA PREPARATION A BIEN ETE ENREGISTREE';
      end;
    4:begin
      s1:='SUPPRESSION IMPOSSIBLE :';
      s2:='PAS DE PREPARATION A SUPPRIMER';
      end;
    5:begin
      s1:='SUPPRESSION TERMINEE :';
      s2:='LA PREPARATION A BIEN ETE SUPPRIMEE';
      end;
    6:begin
```

```
      s1:='MODIFICATION IMPOSSIBLE :';
      s2:='PAS DE PREPARATION A MODIFIER');
      end;
    7:begin
      s1:='MODIFICATIONS TERMINEES :';
      s2:='LES MODIFICATIONS ONT ETE ENREGISTREES';
      end;
    8:begin
      s1:='MODIFICATIONS TERMINEES :';
      s2:='AUCUNE MODIFICATION N EST ENREGISTREE';
      end;
    end;
    gotoxy(0,10);
    write(s1);
    gotoxy(0,12);
    write(s2);
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ord(ch)<>32 do read(keyboard,ch);
end;

procedure ENREGISTRE(var z:integer);
begin
    effecr;
    gotoxy(9,0);
    write('MODIFIER UNE PREPARATION');
    gotoxy(9,1);
    write('========================');
    gotoxy(0,5);
    write('1- ENREGISTREMENT DES MODIFICATIONS');
    gotoxy(0,6);
    write('   ---------------');
    gotoxy(0,9);
    write('2- ANNULATION DES MODIFICATIONS');
    gotoxy(0,10);
    write('   ----------');
    gotoxy(0,14);
    write('VOTRE CHOIX :-');
    gotoxy(0,15);
    write('------------');
    gotoxy(13,14);
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        gotoxy(13,14);
        if ord(ch)=49 then begin
                           write(ch);
                           gotoxy(0,5);
                           z:=1;
                           end
        else if ord(ch)=50 then begin
                                write(ch);
                                gotoxy(0,9);
                                z:=2;
                                end
             else write('-');
        read(keyboard,ch);
    end;
end;
```

```
begin
effecr;
selmode(mo);
while mo <> 6 do
begin
    effecr;
    case mo of
    1:begin
      selex(mo,f);
      effecr;
      listeprepa(f);
      end;
    2:begin
      selex(mo,f);
      effecr;
      exafich(f,bs);
      selprepa(f,a,bs,fini,i);
      if fini=false then begin
                         ouvre(f);
                         lecture(f,i);
                         ferme(f);
                         affiche(f);
                         end;
      end;
    3:begin
      selex(mo,f);
      effecr;
      exafich(f,bs);
      if bs=maxindfich then message(1)
      else begin
           ouvre(f);
           creer(f,fini);
           if fini=true then begin
                             ferme(f);
                             message(2);
                             end

           else begin
                ecriture(f,bs+1);
                ferme(f);
                message(3);
                end;
           end;
      end;

    4:begin
      selex(mo,f);
      effecr;
      exafich(f,bs);
      if bs=0 then message(6)
      else begin
           selprepa(f,m,bs,fini,i);
           if fini=false
           then begin
                ouvre(f);
                lecture(f,i);
                modifier(f,fini);
                if fini=true then begin
                                  ferme(f);
                                  message(8);
                                  end
```

```
                else begin
                     enregistre(q);
                     if q=1 then begin
                                 ecriture(f,i);
                                 ferme(f);
                                 message(7);
                                 end
                     else begin
                          ferme(f);
                          message(8);
                          end;
                     end;
                end;
          end;
     end;
   5:begin
     selex(mo,f);
     effecr;
     exafich(f,bs);
     if bs=0 then message(4)
     else begin
          selprepa(f,s,bs,fini,i);
          if fini=false
          then begin
               if i=bs
               then begin
                    ouvre(f);
                    case f of
                    1:f1^.nomprepa:='                    ';
                    2:f2^.nomprepa:='                    ';
                    3:f3^.nomprepa:='                    ';
                    end;
                    ecriture(f,i);
                    ferme(f);
                    end
               else begin
                    ouvre(f);
                    lecture(f,bs);
                    ecriture(f,i);
                    case f of
                    1:f1^.nomprepa:='                    ';
                    2:f2^.nomprepa:='                    ';
                    3:f3^.nomprepa:='                    ';
                    end;
                    ecriture(f,bs);
                    ferme(f);
                    end;
               message(5);
               end;
          end;
     end;


   end;
   effecr;
   selmode(mo);
end;
end;
```

```
segment procedure AFFICHE(f:nofich);

var ch:char;
     x:coorx;
     y:coory;
     j,k,i:integer;

procedure AFF1(f:nofich);

var reponse:packed array[1..25] of char;

begin
    case f of
    1:reponse:=f1^.nomprepa;
    2:reponse:=f2^.nomprepa;
    3:reponse:=f3^.nomprepa;
    end;
    gotoxy(3,0);
    write('AFFICHAGE : ',reponse);
    gotoxy(3,1);
    write('==========');
    ligne;
    gotoxy(0,23);
    write('(ESC = ARRET DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while (ord(ch)<>32) and
          (ord(ch)<>27) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
end;

procedure AFF2(f:nofich);

var reputil:packed array[1..6] of boolean;

begin
    case f of
    1:for i:=1 to 6 do
      reputil[i]:=f1^.reputil[i];
    2:for i:=1 to 6 do
      reputil[i]:=f2^.reputil[i];
    3:for i:=1 to 6 do
      reputil[i]:=f3^.reputil[i];
    end;
    gotoxy(0,3);
    write('REPRESENTATIONS UTILISEES:');
    gotoxy(0,4);
    write('-------------------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> ');
    gotoxy(0,7);
    write('2-GRAPHE-------> ');
    gotoxy(0,9);
    write('3-ARBRE--------> ');
    gotoxy(0,11);
    write('4-LINEAIRE-----> ');
    gotoxy(0,13);
    write('5-MATH.+.------> ');
    gotoxy(0,15);
```

```
    write('6-MATH.-.-------> ');
    x:=17;
    y:=3;
    for i:=1 to 6 do
    begin
        gotoxy(x,y+(2*i));
        if reputil[i]=true then write('OUI')
                           else write('NON');
    end;
    ligne;
    gotoxy(0,23);
    write('(ESC = ARRET DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while (ord(ch)<>32) and
          (ord(ch)<>27) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
end;


procedure AFF3(f:nofich);


var varspat:packed array[1..6] of boolean;

begin
    case f of
    1:for i:=1 to 6 do
      varspat[i]:=f1^.varspat[i];
    2:for i:=1 to 6 do
      varspat[i]:=f2^.varspat[i];
    3:for i:=1 to 6 do
      varspat[i]:=f3^.varspat[i];
    end;
    gotoxy(0,3);
    write('DISPOSITION SPATIALE VARIABLE:');
    gotoxy(0,4);
    write('------------------------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> ');
    gotoxy(0,7);
    write('2-GRAPHE-------> ');
    gotoxy(0,9);
    write('3-ARBRE--------> ');
    gotoxy(0,11);
    write('4-LINEAIRE-----> ');
    gotoxy(0,13);
    write('5-MATH.+.------> ');
    gotoxy(0,15);
    write('6-MATH.-.------> ');
    x:=17;
    y:=3;
    for i:=1 to 6 do
    begin
        gotoxy(x,y+(2*i));
        if varspat[i]=true then write('OUI')
                           else write('NON');
    end;
    ligne;
    gotoxy(0,23);
```

```
    write('(ESC = ARRET DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while (ord(ch)<>32) and
          (ord(ch)<>27) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
end;


procedure AFF4(f:nofich);


var interve:packed array[1..6] of boolean;

begin
    case f of
    1:for i:=1 to 6 do
      interve[i]:=f1^.interve[i];
    2:for i:=1 to 6 do
      interve[i]:=f2^.interve[i];
    3:for i:=1 to 6 do
      interve[i]:=f3^.interve[i];
    end;

    gotoxy(0,3);
    write('INVERSION DES PARTIES:');
    gotoxy(0,4);
    write('----------------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> ');
    gotoxy(0,7);
    write('2-GRAPHE-------> ');
    gotoxy(0,9);
    write('3-ARBRE--------> ');
    gotoxy(0,11);
    write('4-LINEAIRE-----> ');
    gotoxy(0,13);
    write('5-MATH.+.------> ');
    gotoxy(0,15);
    write('6-MATH.-.------> ');
    x:=17;
    y:=3;
    for i:=1 to 6 do
    begin
        gotoxy(x,y+(2*i));
        if interve[i]=true then write('OUI')
                           else write('NON');
    end;
    ligne;
    gotoxy(0,23);
    write('(ESC = ARRET DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while (ord(ch)<>32) and
          (ord(ch)<>27) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
end;
```

```
procedure AFF6(f:nofich);

var lig,rep:string[9];
    nbrex:packed array[1..6,1..6] of numex;
    nbrex2:packed array[1..5] of numex;


procedure C5ASS;


begin

    gotoxy(0,3);
    write('NOMBRE D EXERCICES:');
    gotoxy(0,4);
    write('------------------');
    gotoxy(0,6);
    write('POUR LES SITUATIONS LIEES PAR:');
    gotoxy(0,8);
    write('1- ASSOCIATION 2-1-1 --------> ');
    gotoxy(0,10);
    write('2- ASSOCIATION 2-2 ----------> ');
    gotoxy(0,12);
    write('3- ASSOCIATION 3-1 ----------> ');
    gotoxy(0,14);
    write('4- ASSOCIATION 4 ------------> ');
    gotoxy(0,16);
    write('5- AUCUNE ASSOCIATION -------> ');
    x:=31;
    y:=6;
    for i:=1 to 5 do
    begin
        gotoxy(x,y+(2*i));
        if nbrex2[i]<10 then write('0',nbrex2[i])
                        else write(nbrex2[i]);
    end;
    ligne;
    gotoxy(0,23);
    write('(ESC = ARRET DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while (ord(ch)<>32) and
          (ord(ch)<>27) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
end;


procedure C5SEL1;

begin
    gotoxy(0,3);
    write('NOMBRE D EXERCICES :');
    gotoxy(0,4);
    write('------------------');
    gotoxy(0,5);
    write('SOIT UNE SITUATION DE DEPART PRESENTEE');
```

```
    gotoxy(0,6);
    write('SOUS FORME DE ');
    gotoxy(0,8);
    write('ET UNE SITUATION A RECONNAITRE');
    gotoxy(0,9);
    write('SOUS FORME DE :                    NBR.EX.:');
    gotoxy(32,10);
    write('--------');
    x:=10;
    y:=11;
    for i:=1 to 6 do
    begin
        gotoxy(x,y+i-1);
        write(i,'-');
        case i of
        1:write('DIAGRAMME----------->');
        2:write('GRAPHE-------------->');
        3:write('ARBRE--------------->');
        4:write('LINEAIRE------------>');
        5:write('MATH.+.------------->');
        6:write('MATH.-.------------->');
        end;
    end;

end;

procedure C5SEL2;

begin
    for i:=1 to 6 do
    begin
        case i of
        1:begin
          rep:='DIAGRAMME';
          lig:='----------';
          end;
        2:begin
          rep:='GRAPHE   ';
          lig:='------   ';
          end;
        3:begin
          rep:='ARBRE    ';
          lig:='-----    ';
          end;
        4:begin
          rep:='LINEAIRE ';
          lig:='-------- ';
          end;
        5:begin
          rep:='MATH.+.  ';
          lig:='-------  ';
          end;
        6:begin
          rep:='MATH.-.  ';
          lig:='-------  ';
          end;
        end;
        gotoxy(14,6);
        write(rep);
        gotoxy(14,7);
        write(lig);
        x:=34;
```

```
        y:=10;
        for k:=1 to 6 do
        begin
        gotoxy(x,y+k);
        if nbrex[i,k]<10 then write('0',nbrex[i,k])
                         else write(nbrex[i,k]);
        end;
        ligne;
        gotoxy(0,23);
        write('(ESC = ARRET DE L AFFICHAGE)');

        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
        while ord(ch)<>32 do
        if ord(ch)=27 then exit(affiche)
                      else read(keyboard,ch);

    end;
end;

begin
    case f of
    1:begin
      for i:=1 to 6 do
      for j:=1 to 6 do
      nbrex[i,j]:=f1^.nbrex[i,j];
      c5sel1;
      c5sel2;
      end;
    2:begin
      for i:=1 to 6 do
      for j:=1 to 6 do
      nbrex[i,j]:=f2^.nbrex[i,j];
      c5sel1;
      c5sel2;
      end;
    3:begin
      for i:=1 to 5 do
      nbrex2[i]:=f3^.nbrex[i];
      c5ass;
      end;
    end;

end;

procedure AFF7(f:nofich) ;


var valmax:10..100;

begin
    case f of
    1:valmax:=f1^.valmax;
    2:valmax:=f2^.valmax;
    3:valmax:=f3^.valmax;
    end;
    gotoxy(0,3);
    write('VALEUR MAXIMALE DE LA SOMME');
    gotoxy(0,4);
    write('----------------------------');
    gotoxy(0,6);
```

```
    write('SOIT A+B=C. LA PLUS GRANDE VALEUR QUE');
    gotoxy(0,8);
    write('C PEUT ATTEINDRE EST ');
    if valmax=100 then write('100')
    else write('0',valmax);
    ligne;
    gotoxy(0,23);
    write('(ESC = ARRET DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while (ord(ch)<>32) and
          (ord(ch)<>27) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
end;


procedure AFF5(f:nofich);


var vardess:packed array[1..2] of boolean;

begin
    case f of
    1:for i:=1 to 2 do
      vardess[i]:=f1^.vardess[i];
    2:for i:=1 to 2 do
      vardess[i]:=f2^.vardess[i];
    3:for i:=1 to 2 do
      vardess[i]:=f3^.vardess[i];
    end;
    gotoxy(0,3);
    write('DESSIN VARIABLE:');
    gotoxy(0,4);
    write('----------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> ');
    if vardess[1]=true then write('OUI')
                       else write('NON');
    gotoxy(0,7);
    write('2-GRAPHE-------> ');
    if vardess[2]=true then write('OUI')
                       else write('NON');
    ligne;
    gotoxy(0,23);
    write('(ESC = ARRET DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while (ord(ch)<>27) and
          (ord(ch)<>32) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
end;
```

```
procedure AFF8(f:nofich);

var passdix:boolean;

begin
    case f of
    1:passdix:=f1^.passdix;
    2:passdix:=f2^.passdix;
    3:passdix:=f3^.passdix;
    end;
    gotoxy(0,3);
    write('REPORT A LA DIZAINE');
    gotoxy(0,4);
    write('--------------------');
    gotoxy(0,6);
    write('LES SITUATIONS OU LE CALCUL DE A+B');
    gotoxy(0,8);
    write('OCCASIONNENT UN REPORT A LA DIZAINE');
    gotoxy(0,10);
    write('SONT ADMISES ------> ');
    if passdix=true then write('OUI')
                    else write('NON');
    ligne;
    gotoxy(0,23);
    write('(ESC = ARRET DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ord(ch)<>32 do
    if ord(ch)=27 then exit(affiche)
                  else read(keyboard,ch);
end;

procedure AFF9 (f:nofich);

var tabsit1:packed array[1..9]of boolean;

procedure DIS1;
begin
    gotoxy(0,6);
    write('-MEMES SYMBOLES:');
    gotoxy(5,8);
    write('1- A+?=B , ?+B=A ---------> ');
    if tabsit1[1]=true then write('OUI')
                       else write('NON');
    gotoxy(5,9);
    write('    (B>A)    (A>B)');
    gotoxy(0,11);
    write('-A OU B REMPLACE PAR C:');
    gotoxy(5,13);
    write('2- C+B=? , A+C=? ---------> ');
    if tabsit1[2]=true then write('OUI')
                       else write('NON');
    gotoxy(5,15);
    write('3- ?+B=C , A+?=C ---------> ');
    if tabsit1[3]=true then write('OUI')
                       else write('NON');
end;
```

```
procedure DIS2;
begin
    gotoxy(0,6);
    write('-A OU B REMPLACE PAR X:');
    gotoxy(5,8);
    write('4- X+B=? , A+X=? ----------> ');
    if tabsit1[4]=true then write('OUI')
                       else write('NON');
    gotoxy(5,10);
    write('5- ?+B=X , A+?=X !');
    gotoxy(5,11);
    write('    (X>B)    (X>A) !');
    gotoxy(5,12);
    write('                    !---------> ');
    if tabsit1[5]=true then write('OUI')
                       else write('NON');
    gotoxy(5,13);
    write('    ?+X=B , X+?=A !');
    gotoxy(5,14);
    write('    (X<B)    (X<A) !');
end;

procedure DIS3;
begin
    gotoxy(0,6);
    write('-A ET B REMPLACES PAR C ET X:');
    gotoxy(5,8);
    write('6- X+C=? ------------------> ');
    if tabsit1[6]=true then write('OUI')
                       else write('NON');
    gotoxy(5,10);
    write('7- ?+C=X , X+?=C ----------> ');
    if tabsit1[7]=true then write('OUI')
                       else write('NON');
    gotoxy(5,11);
    write('    (X>C)    (C>X) ');
    gotoxy(0,13);
    write('-A ET B REMPLACES PAR X ET Y (X>Y):');
    gotoxy(5,15);
    write('8- X+Y=? ------------------> ');
    if tabsit1[8]=true then write('OUI')
                       else write('NON');
    gotoxy(5,17);
    write('9- ?+Y=X ------------------> ');
    if tabsit1[9]=true then write('OUI')
                       else write('NON');
end;



begin
    case f of
    1:for i:=1 to 9 do
      tabsit1[i]:=f1^.tabsit1[i];
    2:for i:=1 to 9 do
      tabsit1[i]:=f2^.tabsit1[i];
    3:for i:=1 to 9 do
      tabsit1[i]:=f3^.tabsit1[i];
    end;
    gotoxy(0,3);
    write('DISTORSIONS PAR RAPPORT A A+B=? SI A+B=C');
    gotoxy(0,4);
```

```
    write('-----------------------------------');
    dis1;
    ligne;
    gotoxy(0,23);
    write('(ESC = FIN DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while (ord(ch)<>27) and
          (ord(ch)<>32) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);

    effecr3;
    dis2;
    gotoxy(26,23);
    read(keyboard,ch);
    while (ord(ch)<>27) and
          (ord(ch)<>32) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
    effecr3;
    dis3;
    gotoxy(26,23);
    read(keyboard,ch);
    while (ord(ch)<>27) and
          (ord(ch)<>32) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);

end;

procedure AFF10 (f:nofich);


var
    tabsit2:packed array[1..9]of boolean;

procedure DIS1;
begin
    gotoxy(0,6);
    write('-MEMES SYMBOLES:');
    gotoxy(5,8);
    write('1- A+C=? ------------------> ');
    if tabsit2[1]=true then write('OUI')
                       else write('NON');
    gotoxy(0,10);
    write('-A OU C REMPLACE PAR B:');
    gotoxy(5,12);
    write('2- C+B=? , A+B=? ----------> ');
    if tabsit2[2]=true then write('OUI')
                       else write('NON');
    gotoxy(5,14);
    write('3- ?+B=C , A+?=B , ?+B=A -> ');
    if tabsit2[3]=true then write('OUI')
                       else write('NON');
    gotoxy(5,15);
    write('              (B>A)    (A>B)');

end;
```

```
procedure DIS2;
begin
    gotoxy(0,6);
    write('-A OU C REMPLACE PAR X:');
    gotoxy(5,8);
    write('4- X+C=? , A+X=? ---------> ');
    if tabsit2[4]=true then write('OUI')
                       else write('NON');
    gotoxy(5,10);
    write('5- ?+C=X , A+?=X !');
    gotoxy(5,11);
    write('    (X>C)    (X>A) !');
    gotoxy(5,12);
    write('                   !--------> ');
    if tabsit2[5]=true then write('OUI')
                       else write('NON');
    gotoxy(5,13);
    write('    ?+X=C , X+?=A !');
    gotoxy(5,14);
    write('    (X<C)    (X<A) !');
end;

procedure DIS3;
begin
    gotoxy(0,6);
    write('-A ET C REMPLACES PAR B ET X:');
    gotoxy(5,8);
    write('6- X+B=? ----------------> ');
    if tabsit2[6]=true then write('OUI')
                       else write('NON');
    gotoxy(5,10);
    write('7- ?+B=X , X+?=B ---------> ');
    if tabsit2[7]=true then write('OUI')
                       else write('NON');
    gotoxy(5,11);
    write('    (X>B)    (B>X) ');
    gotoxy(0,13);
    write('-A ET C REMPLACES PAR X ET Y (X>Y):');
    gotoxy(5,15);
    write('8- X+Y=? ----------------> ');
    if tabsit2[8]=true then write('OUI')
                       else write('NON');
    gotoxy(5,17);
    write('9- ?+Y=X ----------------> ');
    if tabsit2[9]=true then write('OUI')
                       else write('NON');
end;


begin
    case f of
    1:for i:=1 to 9 do
      tabsit2[i]:=f1^.tabsit2[i];
    2:for i:=1 to 9 do
      tabsit2[i]:=f2^.tabsit2[i];
    3:for i:=1 to 9 do
      tabsit2[i]:=f3^.tabsit2[i];
    end;
    gotoxy(0,3);
    write('DISTORSIONS PAR RAPPORT A A+?=C SI A+B=C');
    gotoxy(0,4);
```

```
    write('--------------------------------');
    dis1;
    ligne;
    gotoxy(0,23);
    write('(ESC = FIN DE L AFFICHAGE)');
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while (ord(ch)<>27) and
          (ord(ch)<>32) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
    effecr3;
    dis2;
    gotoxy(26,23);
    read(keyboard,ch);
    while (ord(ch)<>27) and
          (ord(ch)<>32) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);

    effecr3;
    dis3;
    gotoxy(26,23);
    read(keyboard,ch);
    while (ord(ch)<>27) and
          (ord(ch)<>32) do read(keyboard,ch);
    if ord(ch)=27 then exit(affiche);
end;


begin

    effecr;
    aff1(f);
    effecr1;
    aff2(f);
    effecr1;
    aff3(f);
    effecr1;
    aff4(f);
    effecr1;
    aff5(f);
    effecr1;
    aff6(f);
    effecr1;
    aff7(f);
    effecr1;
    aff8(f);
    effecr1;
    aff9(f);
    effecr1;
    aff10(f);
end;
```

```
segment procedure CREER(f:nofich;var fin:boolean);

var ch:char;
     x:coorx;
     y:coory;
     centaine,dizaine,unite,nb,nc,bi,bs,j,k,i:integer;

procedure CREER1(f:nofich);

var reponse:packed array[1..25] of char;

begin
    fin:=false;
    gotoxy(3,0);
    write('CREATION : ');
    gotoxy(3,1);
    write('========');
    ligne;
    gotoxy(0,19);
    write('DONNEZ UN NOM A LA PREPARATION');
    gotoxy(0,20);
    write('           ---      ----------');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');
    reponse:='                         ';
    while reponse='                         ' do
    begin
        gotoxy(0,21);
        write('R:-------------------------');
        gotoxy(2,21);
        i:=0;
        x:=2;
        y:=21;
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
        while not eoln(keyboard) do
        begin
            if (ord(ch) >=32) and
               (ord(ch) <=126) and
               (i<25)
            then begin
                 write(ch);
                 i:=i+1;
                 reponse[i]:=ch;
                 end
            else
                 if (ord(ch)=8) and
                    (i>0)
                 then begin
                      reponse[i]:=' ';
                      i:=i-1;
                      gotoxy(x+i,y);
                      write('-');
                      gotoxy(x+i,y);
                      end
                 else
                      if ord(ch)=27
                      then begin
                           fin:=true;
                           exit(creer);
                           end;
```

```
            read(keyboard,ch);
        end;
    end;
    gotoxy(14,0);
    write(reponse);
    case f of
    1:f1^.nomprepa:=reponse;
    2:f2^.nomprepa:=reponse;
    3:f3^.nomprepa:=reponse;
    end;
    read(keyboard,ch);
    while ch<>chr(32) do read(keyboard,ch);

end;


procedure CREER2(f:nofich);

var reputil:packed array[1..6] of boolean;

begin
    for i:=1 to 6
    do reputil[i]:=false;
    gotoxy(0,3);
    write('REPRESENTATIONS UTILISEES:');
    gotoxy(0,4);
    write('-------------------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> NON');
    gotoxy(0,7);
    write('2-GRAPHE-------> NON');
    gotoxy(0,9);
    write('3-ARBRE--------> NON');
    gotoxy(0,11);
    write('4-LINEAIRE-----> NON');
    gotoxy(0,13);
    write('5-MATH.+.------> NON');
    gotoxy(0,15);
    write('6-MATH.-.------> NON');
    ligne;
    gotoxy(0,19);
    write('QUELLES REPRESENTATIONS UTILISER ?');
    gotoxy(0,20);
    case f of
    1:bs:=1;
    2:bs:=3;
    3:bs:=1;
    end;
    write('              (AU MOINS ',bs,')');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');
    gotoxy(2,21);
    bi:=0;
    fin:=false;
    while (bi<bs) do
    begin
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
        while ord(ch)<>32 do
```

```
        begin
            if ord(ch)=27
            then begin
                 fin:=true;
                 exit(creer);
                 end
            else
                 if (ord(ch)>=49) and
                    (ord(ch)<=54)
                 then begin
                      write(ch);
                      i:=ord(ch)-48;
                      case i of
                      1:gotoxy(17,5);
                      2:gotoxy(17,7);
                      3:gotoxy(17,9);
                      4:gotoxy(17,11);
                      5:gotoxy(17,13);
                      6:gotoxy(17,15);
                      end;
                      if reputil[i] = false
                      then begin
                           reputil[i]:=true;
                           write('OUI');
                           bi:=bi+1;
                           end
                      else begin
                           reputil[i]:=false;
                           write('NON');
                           bi:=bi-1;
                           end;
                      case i of
                      1:gotoxy(0,5);
                      2:gotoxy(0,7);
                      3:gotoxy(0,9);
                      4:gotoxy(0,11);
                      5:gotoxy(0,13);
                      6:gotoxy(0,15);
                      end;

                 end;
            read(keyboard,ch);
            gotoxy(2,21);
            write('-');
            gotoxy(2,21);

            end;
        end;
        case f of
        1:for i:=1 to 6 do
          f1^.reputil[i]:=reputil[i];
        2:for i:=1 to 6 do
          f2^.reputil[i]:=reputil[i];
        3:for i:=1 to 6 do
          f3^.reputil[i]:=reputil[i];
        end;
end;
```

```
procedure CREER3(f:nofich);

var
    varspat:packed array[1..6] of boolean;

begin
    for i:=1 to 6
    do varspat[i]:=false;
    gotoxy(0,3);
    write('DISPOSITION SPATIALE VARIABLE:');
    gotoxy(0,4);
    write('-----------------------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> NON');
    gotoxy(0,7);
    write('2-GRAPHE-------> NON');
    gotoxy(0,9);
    write('3-ARBRE--------> NON');
    gotoxy(0,11);
    write('4-LINEAIRE-----> NON');
    gotoxy(0,13);
    write('5-MATH.+.------> NON');
    gotoxy(0,15);
    write('6-MATH.-.------> NON');
    ligne;
    gotoxy(0,19);
    write('L ORIENTATION DU DESSIN PEUT VARIER.');
    gotoxy(0,20);
    write('POUR QUELLES REPRESENTATIONS ?');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');
    gotoxy(2,21);
    fin:=false;
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27
        then begin
             fin:=true;
             exit(creer);
             end
        else
             if (ord(ch)>=49) and
                (ord(ch)<=54)
             then begin
                  write(ch);
                  i:=ord(ch)-48;
                  case i of
                  1:gotoxy(17,5);
                  2:gotoxy(17,7);
                  3:gotoxy(17,9);
                  4:gotoxy(17,11);
                  5:gotoxy(17,13);
                  6:gotoxy(17,15);
                  end;
                  if varspat[i] = false
                  then begin
```

```
                        varspat[i]:=true;
                        write('OUI');
                        end
                    else begin
                        varspat[i]:=false;
                        write('NON');
                        end;
                    case i of
                    1:gotoxy(0,5);
                    2:gotoxy(0,7);
                    3:gotoxy(0,9);
                    4:gotoxy(0,11);
                    5:gotoxy(0,13);
                    6:gotoxy(0,15);
                    end;

              end;
    read(keyboard,ch);
    gotoxy(2,21);
    write('-');
    gotoxy(2,21);

    end;
    case f of
    1:for i:=1 to 6 do
      f1^.varspat[i]:=varspat[i];
    2:for i:=1 to 6 do
      f2^.varspat[i]:=varspat[i];
    3:for i:=1 to 6 do
      f3^.varspat[i]:=varspat[i];
    end;
end;



procedure CREER4(f:nofich);

var interve:packed array[1..6] of boolean;

begin
    for i:=1 to 6
    do interve[i]:=false;
    gotoxy(0,3);
    write('INVERSION DES PARTIES:');
    gotoxy(0,4);
    write('----------------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> NON');
    gotoxy(0,7);
    write('2-GRAPHE-------> NON');
    gotoxy(0,9);
    write('3-ARBRE--------> NON');
    gotoxy(0,11);
    write('4-LINEAIRE-----> NON');
    gotoxy(0,13);
    write('5-MATH.+.------> NON');
    gotoxy(0,15);
    write('6-MATH.-.------> NON');
    ligne;
    gotoxy(0,19);
```

```
write('SOIT A+B=C ,PEUT-ON INTERVERTIR LES ');
gotoxy(0,20);
write('LOCALISATIONS DE A ET DE B ?');
gotoxy(0,21);
write('POUR QUELLES REPRESENTATIONS ?');
gotoxy(0,22);
write('R:-');
gotoxy(0,23);
write('(ESC = ARRET DE LA CREATION)');
gotoxy(2,22);
fin:=false;
if keypress then read(keyboard,ch);
while not keypress do begin end;
read(keyboard,ch);
while ord(ch)<>32 do
begin
    if ord(ch)=27
    then begin
         fin:=true;
         exit(creer);
         end
    else
         if (ord(ch)>=49) and
            (ord(ch)<=54)
         then begin
              write(ch);
              i:=ord(ch)-48;
              case i of
              1:gotoxy(17,5);
              2:gotoxy(17,7);
              3:gotoxy(17,9);
              4:gotoxy(17,11);
              5:gotoxy(17,13);
              6:gotoxy(17,15);
              end;
              if interve[i] = false
              then begin
                   interve[i]:=true;
                   write('OUI');
                   end
              else begin
                   interve[i]:=false;
                   write('NON');
                   end;
              case i of
              1:gotoxy(0,5);
              2:gotoxy(0,7);
              3:gotoxy(0,9);
              4:gotoxy(0,11);
              5:gotoxy(0,13);
              6:gotoxy(0,15);
              end;

         end;
read(keyboard,ch);
gotoxy(2,22);
write('-');
gotoxy(2,22);

end;
case f of
1:for i:=1 to 6 do
```

```
      f1^.interve[i]:=interve[i];
    2:for i:=1 to 6 do
      f2^.interve[i]:=interve[i];
    3:for i:=1 to 6 do
      f3^.interve[i]:=interve[i];
    end;
end;



procedure CREER6(f:nofich);


var lig,rep:string[9];
    nbrex:packed array[1..6,1..6] of numex;
    nbrex2:packed array[1..5] of numex;

procedure SUITE2;

begin
    effecr2;
    gotoxy(0,19);
    write('INTRODUISEZ UN NOMBRE D EXERCICES');
    gotoxy(0,20);
    write('COMPRIS ENTRE 00 ET 25');
    gotoxy(31,6+(nb*2));
    write('--');
    gotoxy(31,6+(nb*2));
    nc:=0;
    read(keyboard,ch);
    while (nc<2) or (not eoln(keyboard) ) do
    begin
        if (nc=0) and
           (ord(ch)>=48) and
           (ord(ch)<=50)

        then begin
             write(ch);
             dizaine:=(ord(ch)-48)*10;
             nc:=nc+1;
             end

        else if (nc=1) and
                (ord(ch)>=48) and
                (ord(ch)<=53) and
                (dizaine=20)
             then begin
                  write(ch);
                  unite:=ord(ch)-48;
                  nc:=nc+1;
                  end

             else if (nc=1) and
                     (dizaine<>20) and
                     (ord(ch)>=48) and
                     (ord(ch)<=57)
                  then begin
                       write(ch);
                       unite:=ord(ch)-48;
                       nc:=nc+1;
                       end
                   else if (nc>0) and
```

```
                        (ord(ch)=8)

                    then begin
                         gotoxy(30+nc,6+(nb*2));
                         write('-');
                         gotoxy(30+nc,6+(nb*2));
                         nc:=nc-1;
                         end;

        read(keyboard,ch);

    end;
    nbrex2[nb]:=dizaine+unite;
    effecr2;
    gotoxy(0,19);
    write('GENRE D ASSOCIATION ?');
    gotoxy(0,21);
    write('R:');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');

end;


procedure C5ASS;



begin
    fin:=false;
    for i:=1 to 5 do
    nbrex2[i]:=0;
    gotoxy(0,3);
    write('NOMBRE D EXERCICES:');
    gotoxy(0,4);
    write('-------------------');
    gotoxy(0,6);
    write('POUR LES SITUATIONS LIEES PAR:');
    gotoxy(0,8);
    write('1- ASSOCIATION 2-1-1 --------> 00');
    gotoxy(0,10);
    write('2- ASSOCIATION 2-2 ----------> 00');
    gotoxy(0,12);
    write('3- ASSOCIATION 3-1 ----------> 00');
    gotoxy(0,14);
    write('4- ASSOCIATION 4 ------------> 00');
    gotoxy(0,16);
    write('5- AUCUNE ASSOCIATION -------> 00');
    ligne;
    gotoxy(0,19);
    write('GENRE D ASSOCIATION ?');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');
    gotoxy(2,21);
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27 then begin
```

```
                              fin:=true;
                              exit(creer);
                              end
        else if (ord(ch)>=49) and
                (ord(ch)<=53)
             then begin
                  gotoxy(2,21);
                  write(ch);
                  nb:=ord(ch)-48;
                  gotoxy(0,6+(nb*2));
                  read(keyboard,ch);
                  while (ord(ch)>=49) and
                        (ord(ch)<=53) do
                  begin
                      gotoxy(2,21);
                      write(ch);
                      nb:=ord(ch)-48;
                      gotoxy(0,6+(nb*2));
                      read(keyboard,ch);
                  end;
                  if ord(ch)=32 then suite2 ;
                  end;
             gotoxy(2,21);
             write('-');
             gotoxy(2,21);
             read(keyboard,ch);
      end;
      for i:=1 to 5 do
      f3^.nbrex[i]:=nbrex2[i];
end;


procedure SUITE1;

begin
     effecr2;
     gotoxy(0,19);
     write('INTRODUISEZ UN NOMBRE D EXERCICES');
     gotoxy(0,20);
     write('COMPRIS ENTRE 00 ET 25');
     gotoxy(x+25,y+nb-1);
     write('--');
     gotoxy(x+25,y+nb-1);
     nc:=0;
     read(keyboard,ch);
     while (nc<2) or (not eoln(keyboard) ) do
     begin
         if (nc=0) and
            (ord(ch)>=48) and
            (ord(ch)<=50)

         then begin
              write(ch);
              dizaine:=(ord(ch)-48)*10;
              nc:=nc+1;
              end

         else if (nc=1) and
                 (ord(ch)>=48) and
                 (ord(ch)<=53) and
                 (dizaine=20)
```

```
            then begin
                 write(ch);
                 unite:=ord(ch)-48;
                 nc:=nc+1;
                 end

            else if (nc=1) and
                    (dizaine<>20) and
                    (ord(ch)>=48) and
                    (ord(ch)<=57)
                 then begin
                      write(ch);
                      unite:=ord(ch)-48;
                      nc:=nc+1;
                      end

                 else if (nc>0) and
                         (ord(ch)=8)

                      then begin
                           gotoxy(x+24+nc,y+nb-1);
                           write('-');
                           gotoxy(x+24+nc,y+nb-1);
                           nc:=nc-1;
                           end;

        read(keyboard,ch);

    end;
    nbrex[i,nb]:=dizaine+unite;
    effecr2;
    gotoxy(0,19);
    write('NUMERO DE REPRESENTATION ?');
    gotoxy(0,21);
    write('R:');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');

end;



procedure C5SEL1;

begin
    fin:=false;
    for i:=1 to 6 do
        for j:=1 to 6 do
            nbrex[i,j]:=0;

    gotoxy(0,3);
    write('NOMBRE D EXERCICES :');
    gotoxy(0,4);
    write('--------------------');
    gotoxy(0,5);
    write('SOIT UNE SITUATION DE DEPART PRESENTEE');
    gotoxy(0,6);
    write('SOUS FORME DE ');
    gotoxy(0,8);
    write('ET UNE SITUATION A RECONNAITRE');
    gotoxy(0,9);
```

```
    write('SOUS FORME DE :                    NBR.EX.:');
    gotoxy(32,10);
    write('--------');
    x:=10;
    y:=11;
    for i:=1 to 6 do
    begin
        gotoxy(x,y+i-1);
        write(i,'-');
        case i of
        1:write('DIAGRAMME----------->');
        2:write('GRAPHE-------------->');
        3:write('ARBRE--------------->');
        4:write('LINEAIRE------------>');
        5:write('MATH.+.------------->');
        6:write('MATH.-.------------->');
        end;
    end;
    ligne;
end;

procedure C5SEL2;

begin
    for i:=1 to 6 do
    begin
        case i of
        1:begin
          rep:='DIAGRAMME';
          lig:='----------';
          end;
        2:begin
          rep:='GRAPHE    ';
          lig:='------    ';
          end;
        3:begin
          rep:='ARBRE     ';
          lig:='-----     ';
          end;
        4:begin
          rep:='LINEAIRE  ';
          lig:='--------- ';
          end;
        5:begin
          rep:='MATH.+.   ';
          lig:='-------   ';
          end;
        6:begin
          rep:='MATH.-.   ';
          lig:='-------   ';
          end;
        end;
        gotoxy(14,6);
        write(rep);
        gotoxy(14,7);
        write(lig);

        for k:=1 to 6 do
        begin
            gotoxy(35,y+k-1);
            write('00');
        end;
```

```
        gotoxy(0,19);
        write('NUMERO DE REPRESENTATION ?');
        gotoxy(0,21);
        write('R:-');
        gotoxy(0,23);
        write('(ESC = ARRET DE LA CREATION)');
        gotoxy(2,21);

        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);

        while ord(ch)<>32 do
        begin
            if ord(ch)=27 then begin
                                fin:=true;
                                exit(creer);
                                end
            else if (ord(ch)>=49) and
                    (ord(ch)<=54)

                 then begin
                      nb:=ord(ch)-48;
                      write(ch);
                      gotoxy(x,y+nb-1);
                      read(keyboard,ch);
                      while (ord(ch)>=49) and
                            (ord(ch)<=54) do
                      begin
                          gotoxy(2,21);
                          write(ch);
                          nb:=ord(ch)-48;
                          gotoxy(x,y+nb-1);
                          read(keyboard,ch);
                      end;
                      if ord(ch)=32 then suite1;
                      end;
            gotoxy(2,21);
            write('-');
            gotoxy(2,21);
            read(keyboard,ch);
        end;

    end;
    case f of
    1:for i:=1 to 6 do
      for j:=1 to 6 do
      f1^.nbrex[i,j]:=nbrex[i,j];
    2:for i:=1 to 6 do
      for j:=1 to 6 do
      f2^.nbrex[i,j]:=nbrex[i,j];
    end;
end;

begin
    case f of
    1:begin
      c5sel1;
      c5sel2;
      end;
    2:begin
      c5sel1;
```

```
      c5sel2;
      end;
    3:c5ass;
    end;

end;

procedure CREER7(f:nofich) ;


var valmax:10..100;

begin
    fin:=false;
    gotoxy(0,3);
    write('VALEUR MAXIMALE DE LA SOMME');
    gotoxy(0,4);
    write('---------------------------');
    gotoxy(0,6);
    write('SOIT A+B=C. LA PLUS GRANDE VALEUR QUE');
    gotoxy(0,8);
    write('C PEUT ATTEINDRE EST ');
    ligne;
    gotoxy(0,19);
    write('VALEUR MAXIMALE DE LA SOMME ?');
    gotoxy(0,20);
    write('COMPRISE ENTRE 10 ET 100');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');
    x:=21;
    y:=8;
    gotoxy(x,y);
    write('---');
    gotoxy(x,y);
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    nb:=0;
    centaine:=0;
    dizaine:=0;
    unite:=0;

    while (nb<3) or
          (ord(ch)<>32) do
    begin

    if ord(ch)=27 then begin
                       fin:=true;
                       exit(creer);
                       end
    else if (nb=0) and
            ((ord(ch)=48) or (ord(ch)=49))
         then begin
              write(ch);
              nb:=nb+1;
              centaine:=ord(ch)-48;
              end
         else if (nb=1) and
                 (centaine=1) and
                 (ord(ch)=48)
              then begin
                   write(ch);
```

```
                    dizaine:=0;
                    nb:=nb+1;
                    end
               else if (nb=1) and
                       (centaine=0) and
                       (ord(ch)>=49) and
                       (ord(ch)<=57)
                    then begin
                         write(ch);
                         dizaine:=ord(ch)-48;
                         nb:=nb+1;
                         end
                    else if (nb=2) and
                         (centaine=1) and
                         (ord(ch)=48)
                         then begin
                              write(ch);
                              nb:=nb+1;
                              unite:=0;
                              end
                         else if (nb=2) and
                                 (centaine=0) and
                                 (ord(ch)>=48) and
                                 (ord(ch)<=57)
                              then begin
                                   write(ch);
                                   nb:=nb+1;
                                   unite:=ord(ch)-48;
                                   end
                              else if (nb<=3) and
                                      (nb>=1) and
                                      (ord(ch)=8)
                                   then begin
                                        nb:=nb-1;
                                        gotoxy(x+nb,y);
                                        write('-');
                                        gotoxy(x+nb,y);
                                        end;

    read(keyboard,ch);
    end;
    valmax:=(centaine*100)+
            (dizaine*10)+
            (unite);
    case f of
    1:f1^.valmax:=valmax;
    2:f2^.valmax:=valmax;
    3:f3^.valmax:=valmax;
    end;
end;
```

```
procedure CREER5(f:nofich);

var vardess:packed array[1..2] of boolean;

begin
    for i:=1 to 2
    do vardess[i]:=false;
    gotoxy(0,3);
    write('DESSIN VARIABLE:');
    gotoxy(0,4);
    write('----------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> NON');
    gotoxy(0,7);
    write('2-GRAPHE-------> NON');
    ligne;
    gotoxy(0,19);
    write('UNE MEME REPRESENTATION PEUT-ELLE ETRE');
    gotoxy(0,20);
    write('DESSINEE DE DIFFERENTES MANIERES ?');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');
    gotoxy(2,21);
    fin:=false;
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27
        then begin
             fin:=true;
             exit(creer);
             end
        else
             if (ord(ch)>=49) and
                (ord(ch)<=50)
             then begin
                  write(ch);
                  i:=ord(ch)-48;
                  case i of
                  1:gotoxy(17,5);
                  2:gotoxy(17,7);
                  end;
                  if vardess[i] = false
                  then begin
                       vardess[i]:=true;
                       write('OUI');
                           end
                  else begin
                       vardess[i]:=false;
                       write('NON');
                       end;
                  case i of
                  1:gotoxy(0,5);
                  2:gotoxy(0,7);
                  end;

             end;
```

```
    read(keyboard,ch);
    gotoxy(2,21);
    write('-');
    gotoxy(2,21);

    end;

    case f of
    1:for i:=1 to 2 do
      f1^.vardess[i]:=vardess[i];
    2:for i:=1 to 2 do
      f2^.vardess[i]:=vardess[i];
    3:for i:=1 to 2 do
      f3^.vardess[i]:=vardess[i];
    end;
end;



procedure CREER8(f:nofich);


var passdix:boolean;

begin
    fin:=false;
    gotoxy(0,3);
    write('REPORT A LA DIZAINE');
    gotoxy(0,4);
    write('--------------------');
    gotoxy(0,6);
    write('LES SITUATIONS OU LE CALCUL DE A+B');
    gotoxy(0,8);
    write('OCCASIONNENT UN REPORT A LA DIZAINE');
    gotoxy(0,10);
    write('SONT ADMISES ------> OUI');
    gotoxy(21,11);
    write('---');
    ligne;
    gotoxy(0,19);
    write('REPORT A LA DIZAINE ADMIS ?');
    gotoxy(0,20);
    write('        (O=OUI N=NON)');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');
    gotoxy(2,21);
    passdix:=true;
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27 then begin
                           fin:=true;
                           exit(creer);
                           end
        else if ord(ch)=79
             then begin
                  write(ch);
```

```
                    gotoxy(21,10);
                    write('OUI');
                    gotoxy(2,21);
                    passdix:=true;
                    end
               else if ord(ch)=78
                    then begin
                         write(ch);
                         gotoxy(21,10);
                         write('NON');
                         gotoxy(2,21);
                         passdix:=false;
                         end
                    else begin
                         write('-');
                         gotoxy(2,21);
                         end;
     read(keyboard,ch);

     end;
     case f of
     1:f1^.passdix:=passdix;
     2:f2^.passdix:=passdix;
     3:f3^.passdix:=passdix;
     end;

end;

procedure CREER9 (f:nofich);


var tabsit1:packed array[1..9]of boolean;

procedure DIS1;
begin
     gotoxy(0,6);
     write('-MEMES SYMBOLES:');
     gotoxy(5,8);
     write('1- A+?=B , ?+B=A ----------> NON');
     gotoxy(5,9);
     write('    (B>A)    (A>B)');
     gotoxy(0,11);
     write('-A OU B REMPLACE PAR C:');
     gotoxy(5,13);
     write('2- C+B=? , A+C=? ----------> NON');
     gotoxy(5,15);
     write('3- ?+B=C , A+?=C ----------> NON');
end;

procedure DIS2;
begin
     gotoxy(0,6);
     write('-A OU B REMPLACE PAR X:');
     gotoxy(5,8);
     write('4- X+B=? , A+X=? ----------> NON');
     gotoxy(5,10);
     write('5- ?+B=X , A+?=X !');
     gotoxy(5,11);
     write('    (X>B)    (X>A) !');
     gotoxy(5,12);
     write('                   !---------> NON');
     gotoxy(5,13);
```

```
    write('   ?+X=B , X+?=A !');
    gotoxy(5,14);
    write('   (X<B)   (X<A) !');
end;

procedure DIS3;
begin
    gotoxy(0,6);
    write('-A ET B REMPLACES PAR C ET X:');
    gotoxy(5,8);
    write('6- X+C=? ----------------> NON');
    gotoxy(5,10);
    write('7- ?+C=X , X+?=C ----------> NON');
    gotoxy(5,11);
    write('   (X>C)   (C>X) ');
    gotoxy(0,13);
    write('-A ET B REMPLACES PAR X ET Y (X>Y):');
    gotoxy(5,15);
    write('8- X+Y=? ----------------> NON');
    gotoxy(5,17);
    write('9- ?+Y=X ----------------> NON');
end;


procedure DC8;

begin
    for i:=1 to 9 do tabsit1[i]:=false;
    fin:=false;
    nc:=0;
    gotoxy(0,3);
    write('DISTORSIONS PAR RAPPORT A A+B=? SI A+B=C');
    gotoxy(0,4);
    write('-------------------------------');
    dis1;
    ligne;
    gotoxy(0,19);
    write('SELECTIONNEZ LES DISTORSIONS ADMISES');
    gotoxy(0,20);
    write('       (AU MOINS 2 PARMI 9 AU TOTAL)');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');
    gotoxy(2,21);
end;

begin
    dc8;
    while nc<2 do
    begin
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27 then begin
                           fin:=true;
                           exit(creer);
                           end
        else if (ord(ch)>=49) and
                (ord(ch)<=51)
```

```
          then begin
               nb:=ord(ch)-48;
               gotoxy(2,21);
               write(ch);
               case nb of
               1:gotoxy(33,8);
               2:gotoxy(33,13);
               3:gotoxy(33,15);
               end;
               if tabsit1[nb]=false
               then begin
                    tabsit1[nb]:=true;
                    write('OUI');
                    nc:=nc+1;
                    end
               else begin
                    tabsit1[nb]:=false;
                    write('NON');
                    nc:=nc-1;
                    end;
               case nb of
               1:gotoxy(5,8);
               2:gotoxy(5,13);
               3:gotoxy(5,15);
               end;
               end
          else begin
               gotoxy(2,21);
               write('-');
               gotoxy(2,21);
               end;

     read(keyboard,ch);
end;
     effecr3;
     dis2;
     gotoxy(2,21);
     write('-');
     gotoxy(2,21);
     read(keyboard,ch);
     while ord(ch)<>32 do
     begin
          if ord(ch)=27 then begin
                             fin:=true;
                             exit(creer);
                             end
          else if (ord(ch)>=52) and
                  (ord(ch)<=53)

               then begin
                    nb:=ord(ch)-48;
                    gotoxy(2,21);
                    write(ch);
                    case nb of
                    4:gotoxy(33,8);
                    5:gotoxy(33,12);
                    end;
                    if tabsit1[nb]=false
                    then begin
                         tabsit1[nb]:=true;
                         write('OUI');
```

```
                    nc:=nc+1;
                    end
               else begin
                    tabsit1[nb]:=false;
                    write('NON');
                    nc:=nc-1;
                    end;
               case nb of
               4:gotoxy(5,8);
               5:gotoxy(5,10);
               end;
               end

          else begin
               gotoxy(2,21);
               write('-');
               gotoxy(2,21);
               end;

    read(keyboard,ch);
end;

effecr3;
dis3;
gotoxy(2,21);
write('-');
gotoxy(2,21);
read(keyboard,ch);
while ord(ch)<>32 do
begin
    if ord(ch)=27 then begin
                       fin:=true;
                       exit(creer);
                       end
    else if (ord(ch)>=54) and
            (ord(ch)<=57)

         then begin
               nb:=ord(ch)-48;
               gotoxy(2,21);
               write(ch);
               case nb of
               6:gotoxy(33,8);
               7:gotoxy(33,10);
               8:gotoxy(33,15);
               9:gotoxy(33,17);
               end;
               if tabsit1[nb]=false
               then begin
                    tabsit1[nb]:=true;
                    write('OUI');
                    nc:=nc+1;
                    end
               else begin
                    tabsit1[nb]:=false;
                    write('NON');
                    nc:=nc-1;
                    end;
               case nb of
               6:gotoxy(5,8);
               7:gotoxy(5,10);
               8:gotoxy(5,15);
```

```
                    9:gotoxy(5,17);
                    end;
                    end
                else begin
                    gotoxy(2,21);
                    write('-');
                    gotoxy(2,21);
                    end;

            read(keyboard,ch);
        end;

        if nc<2 then begin
                        effecr3;
                        dis1;
                        gotoxy(2,21);
                        write('-');
                        gotoxy(2,21);
                        end;
    end;
    case f of
    1:for i:=1 to 9 do
      f1^.tabsit1[i]:=tabsit1[i];
    2:for i:=1 to 9 do
      f2^.tabsit1[i]:=tabsit1[i];
    3:for i:=1 to 9 do
      f3^.tabsit1[i]:=tabsit1[i];
    end;
end;



procedure CREER10 (f:nofich);


var
    tabsit2:packed array[1..9]of boolean;

procedure DIS1;
begin
    gotoxy(0,6);
    write('-MEMES SYMBOLES:');
    gotoxy(5,8);
    write('1- A+C=? ------------------> NON');
    gotoxy(0,10);
    write('-A OU C REMPLACE PAR B:');
    gotoxy(5,12);
    write('2- C+B=? , A+B=? ---------> NON');
    gotoxy(5,14);
    write('3- ?+B=C , A+?=B , ?+B=A -> NON');
    gotoxy(5,15);
    write('             (B>A)    (A>B)');

end;

procedure DIS2;
begin
    gotoxy(0,6);
    write('-A OU C REMPLACE PAR X:');
    gotoxy(5,8);
    write('4- X+C=? , A+X=? ---------> NON');
    gotoxy(5,10);
```

```
    write('5- ?+C=X , A+?=X !');
    gotoxy(5,11);
    write('   (X>C)   (X>A) !');
    gotoxy(5,12);
    write('                  !--------> NON');
    gotoxy(5,13);
    write('   ?+X=C , X+?=A !');
    gotoxy(5,14);
    write('   (X<C)   (X<A) !');
end;

procedure DIS3;
begin
    gotoxy(0,6);
    write('-A ET C REMPLACES PAR B ET X:');
    gotoxy(5,8);
    write('6- X+B=? ----------------> NON');
    gotoxy(5,10);
    write('7- ?+B=X , X+?=B ---------> NON');
    gotoxy(5,11);
    write('   (X>B)   (B>X) ');
    gotoxy(0,13);
    write('-A ET C REMPLACES PAR X ET Y (X>Y):');
    gotoxy(5,15);
    write('8- X+Y=? ----------------> NON');
    gotoxy(5,17);
    write('9- ?+Y=X ----------------> NON');
end;


procedure DC9;

begin
    for i:=1 to 9 do tabsit2[i]:=false;
    fin:=false;
    nc:=0;
    gotoxy(0,3);
    write('DISTORSIONS PAR RAPPORT A A+?=C SI A+B=C');
    gotoxy(0,4);
    write('--------------------------------');
    dis1;
    ligne;
    gotoxy(0,19);
    write('SELECTIONNEZ LES DISTORSIONS ADMISES');
    gotoxy(0,20);
    write('        (AU MOINS 2 PARMI 9 AU TOTAL)');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA CREATION)');
    gotoxy(2,21);
end;

begin
    dc9;
    while nc<2 do
    begin
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
    while ord(ch)<>32 do
    begin
```

```
      if ord(ch)=27 then begin
                          fin:=true;
                          exit(creer);
                          end
      else if (ord(ch)>=49) and
              (ord(ch)<=51)

           then begin
                nb:=ord(ch)-48;
                gotoxy(2,21);
                write(ch);
                case nb of
                1:gotoxy(33,8);
                2:gotoxy(33,12);
                3:gotoxy(33,14);
                end;
                if tabsit2[nb]=false
                then begin
                     tabsit2[nb]:=true;
                     write('OUI');
                     nc:=nc+1;
                     end
                else begin
                     tabsit2[nb]:=false;
                     write('NON');
                     nc:=nc-1;
                     end;
                case nb of
                1:gotoxy(5,8);
                2:gotoxy(5,12);
                3:gotoxy(5,14);
                end;
                end
           else begin
                gotoxy(2,21);
                write('-');
                gotoxy(2,21);
                end;

      read(keyboard,ch);
end;
      effecr3;
      dis2;
      gotoxy(2,21);
      write('-');
      gotoxy(2,21);
      read(keyboard,ch);
      while ord(ch)<>32 do
      begin
          if ord(ch)=27 then begin
                              fin:=true;
                              exit(creer);
                              end
          else if (ord(ch)>=52) and
                  (ord(ch)<=53)

               then begin
                    nb:=ord(ch)-48;
                    gotoxy(2,21);
                    write(ch);
                    case nb of
                    4:gotoxy(33,8);
```

```
                    5:gotoxy(33,12);
                    end;
                    if tabsit2[nb]=false
                    then begin
                         tabsit2[nb]:=true;
                         write('OUI');
                         nc:=nc+1;
                         end
                    else begin
                         tabsit2[nb]:=false;
                         write('NON');
                         nc:=nc-1;
                         end;
                    case nb of
                    4:gotoxy(5,8);
                    5:gotoxy(5,10);
                    end;
                    end

               else begin
                    gotoxy(2,21);
                    write('-');
                    gotoxy(2,21);
                    end;

     read(keyboard,ch);
end;

effecr3;
dis3;
gotoxy(2,21);
write('-');
gotoxy(2,21);
read(keyboard,ch);
while ord(ch)<>32 do
begin
     if ord(ch)=27 then begin
                         fin:=true;
                         exit(creer);
                         end
     else if (ord(ch)>=54) and
             (ord(ch)<=57)

          then begin
               nb:=ord(ch)-48;
               gotoxy(2,21);
               write(ch);
               case nb of
               6:gotoxy(33,8);
               7:gotoxy(33,10);
               8:gotoxy(33,15);
               9:gotoxy(33,17);
               end;
               if tabsit2[nb]=false
               then begin
                    tabsit2[nb]:=true;
                    write('OUI');
                    nc:=nc+1;
                    end
               else begin
                    tabsit2[nb]:=false;
                    write('NON');
```

```
                        nc:=nc-1;
                        end;
                    case nb of
                    6:gotoxy(5,8);
                    7:gotoxy(5,10);
                    8:gotoxy(5,15);
                    9:gotoxy(5,17);
                    end;
                    end
                else begin
                    gotoxy(2,21);
                    write('-');
                    gotoxy(2,21);
                    end;

          read(keyboard,ch);
        end;

        if nc<2 then begin
                        effecr3;
                        dis1;
                        gotoxy(2,21);
                        write('-');
                        gotoxy(2,21);
                        end;
    end;
    case f of
    1:for i:=1 to 9 do
      f1^.tabsit2[i]:=tabsit2[i];
    2:for i:=1 to 9 do
      f2^.tabsit2[i]:=tabsit2[i];
    3:for i:=1 to 9 do
      f3^.tabsit2[i]:=tabsit2[i];
    end;
end;

begin
    effecr;
    creer1(f);
    effecr1;
    creer2(f);
    effecr1;
    creer3(f);
    effecr1;
    creer4(f);
    effecr1;
    creer5(f);
    effecr1;
    creer6(f);
    effecr1;
    creer7(f);
    effecr1;
    creer8(f);
    effecr1;
    creer9(f);
    effecr1;
    creer10(f);

end;
```

```
segment procedure MODIFIER (f:nofich;var fin:boolean);

var ch:char;
     x:coorx;
     y:coory;
     centaine,dizaine,unite,nb,nc,bi,bs,j,k,i:integer;

procedure MOD1(f:nofich);

var reponse:packed array[1..25] of char;

begin
    fin:=false;
    case f of
    1:reponse:=f1^.nomprepa;
    2:reponse:=f2^.nomprepa;
    3:reponse:=f3^.nomprepa;
    end;
    gotoxy(0,0);
    write('MODIFICATION : ',reponse);
    gotoxy(0,1);
    write('============');
    ligne;
    gotoxy(0,19);
    write('DONNEZ UN NOM A LA PREPARATION');
    gotoxy(0,20);
    write('           ---       ----------');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA MODIFICATION)');
    reponse:='                         ';
    gotoxy(0,21);
    write('R:------------------------');
    gotoxy(2,21);
    i:=0;
    x:=2;
    y:=21;
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while not eoln(keyboard) do
    begin
        if (ord(ch) >=32) and
           (ord(ch) <=126) and
           (i<25)
        then begin
             write(ch);
             i:=i+1;
             reponse[i]:=ch;
             end
        else
             if (ord(ch)=8) and
                (i>0)
             then begin
                  reponse[i]:=' ';
                  i:=i-1;
                  gotoxy(x+i,y);
                  write('-');
                  gotoxy(x+i,y);
                  end
             else
                  if ord(ch)=27
                  then begin
```

```
                        fin:=true;
                        exit(modifier);
                        end;
        read(keyboard,ch);
    end;
    if reponse<>''
    then begin
         gotoxy(15,0);
         write(reponse);
         case f of
         1:f1^.nomprepa:=reponse;
         2:f2^.nomprepa:=reponse;
         3:f3^.nomprepa:=reponse;
         end;
         read(keyboard,ch);
         while ch<>chr(32) do read(keyboard,ch);
         end;
end;


procedure MOD2(f:nofich);

var reputil:packed array[1..6] of boolean;

procedure SMOD2;
begin
    case f of
    1:for i:=1 to 6 do
      f1^.reputil[i]:=reputil[i];
    2:for i:=1 to 6 do
      f2^.reputil[i]:=reputil[i];
    3:for i:=1 to 6 do
      f3^.reputil[i]:=reputil[i];
    end;
end;
begin
    fin:=false;
    case f of
    1:for i:=1 to 6
      do reputil[i]:=f1^.reputil[i];
    2:for i:=1 to 6
      do reputil[1]:=f2^.reputil[i];
    3:for i:=1 to 6
      do reputil[i]:=f3^.reputil[i];
    end;
    gotoxy(0,3);
    write('REPRESENTATIONS UTILISEES:');
    gotoxy(0,4);
    write('-------------------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> ');
    gotoxy(0,7);
    write('2-GRAPHE-------> ');
    gotoxy(0,9);
    write('3-ARBRE--------> ');
    gotoxy(0,11);
    write('4-LINEAIRE-----> ');
    gotoxy(0,13);
    write('5-MATH.+.------> ');
    gotoxy(0,15);
    write('6-MATH.-.------> ');
    x:=17;
```

```
y:=3;
bi:=0;
for i:=1 to 6 do
begin
    gotoxy(x,y+(2*i));
    if reputil[i]=true then begin
                            write('OUI');
                            bi:=bi+1;
                            end
                       else write('NON');
end;
ligne;
gotoxy(0,19);
write('QUELLES REPRESENTATIONS UTILISER ?');
gotoxy(0,20);
case f of
1:bs:=1;
2:bs:=3;
3:bs:=1;
end;
write('              (AU MOINS ',bs,')');
gotoxy(0,21);
write('R:-');
gotoxy(0,23);
write('(ESC = ARRET DE LA MODIFICATION)');
gotoxy(2,21);
if keypress then read(keyboard,ch);
while not keypress do begin end;
read(keyboard,ch);
while (ord(ch)<>32) or (bi<bs) do
begin
    if ord(ch)=27
    then begin
         fin:=true;
         exit(modifier);
         end
    else
         if (ord(ch)>=49) and
            (ord(ch)<=54)
         then begin
              write(ch);
              i:=ord(ch)-48;
              case i of
              1:gotoxy(17,5);
              2:gotoxy(17,7);
              3:gotoxy(17,9);
              4:gotoxy(17,11);
              5:gotoxy(17,13);
              6:gotoxy(17,15);
              end;
              if reputil[i] = false
              then begin
                   reputil[i]:=true;
                   write('OUI');
                   bi:=bi+1;
                   end
              else begin
                   reputil[i]:=false;
                   write('NON');
                   bi:=bi-1;
                   end;
              case i of
```

```
                    1:gotoxy(0,5);
                    2:gotoxy(0,7);
                    3:gotoxy(0,9);
                    4:gotoxy(0,11);
                    5:gotoxy(0,13);
                    6:gotoxy(0,15);
                    end;

               end;
     read(keyboard,ch);
     gotoxy(2,21);
     write('-');
     gotoxy(2,21);

     end;
     smod2;

end;



procedure MOD3(f:nofich);


var
     varspat:packed array[1..6] of boolean;

procedure SMOD3;
begin
     case f of
     1:for i:=1 to 6 do
       f1^.varspat[i]:=varspat[i];
     2:for i:=1 to 6 do
       f2^.varspat[i]:=varspat[i];
     3:for i:=1 to 6 do
       f3^.varspat[i]:=varspat[i];
     end;
end;

begin
     fin:=false;
     case f of
     1:for i:=1 to 6 do
       varspat[i]:=f1^.varspat[i];
     2:for i:=1 to 6 do
       varspat[i]:=f2^.varspat[i];
     3:for i:=1 to 6 do
       varspat[i]:=f3^.varspat[i];
     end;
     gotoxy(0,3);
     write('DISPOSITION SPATIALE VARIABLE:');
     gotoxy(0,4);
     write('-----------------------------');
     gotoxy(0,5);
     write('1-DIAGRAMME----> ');
     gotoxy(0,7);
     write('2-GRAPHE-------> ');
     gotoxy(0,9);
     write('3-ARBRE--------> ');
     gotoxy(0,11);
     write('4-LINEAIRE-----> ');
     gotoxy(0,13);
```

```
write('5-MATH.+.------> ');
gotoxy(0,15);
write('6-MATH.-.------> ');
x:=17;
y:=3;
for i:=1 to 6 do
begin
    gotoxy(x,y+(2*i));
    if varspat[i]=true then write('OUI')
                       else write('NON');
end;
ligne;
gotoxy(0,19);
write('L ORIENTATION DU DESSIN PEUT VARIER.');
gotoxy(0,20);
write('POUR QUELLES REPRESENTATIONS ?');
gotoxy(0,21);
write('R:-');
gotoxy(0,23);
write('(ESC = ARRET DE LA MODIFICATION)');
gotoxy(2,21);
if keypress then read(keyboard,ch);
while not keypress do begin end;
read(keyboard,ch);
while ord(ch)<>32 do
begin
    if ord(ch)=27
    then begin
         fin:=true;
         exit(modifier);
         end
    else
         if (ord(ch)>=49) and
            (ord(ch)<=54)
         then begin
              write(ch);
              i:=ord(ch)-48;
              case i of
              1:gotoxy(17,5);
              2:gotoxy(17,7);
              3:gotoxy(17,9);
              4:gotoxy(17,11);
              5:gotoxy(17,13);
              6:gotoxy(17,15);
              end;
              if varspat[i] = false
              then begin
                   varspat[i]:=true;
                   write('OUI');
                   end
              else begin
                   varspat[i]:=false;
                   write('NON');
                   end;
              case i of
              1:gotoxy(0,5);
              2:gotoxy(0,7);
              3:gotoxy(0,9);
              4:gotoxy(0,11);
              5:gotoxy(0,13);
              6:gotoxy(0,15);
              end;
```

```
            end;
    read(keyboard,ch);
    gotoxy(2,21);
    write('-');
    gotoxy(2,21);

    end;
    smod3;
end;



procedure MOD4(f:nofich);


var interve:packed array[1..6] of boolean;

procedure SMOD4;
begin
    case f of
    1:for i:=1 to 6 do
      f1^.interve[i]:=interve[i];
    2:for i:=1 to 6 do
      f2^.interve[i]:=interve[i];
    3:for i:=1 to 6 do
      f3^.interve[i]:=interve[i];
    end;
end;

begin
    fin:=false;
    case f of
    1:for i:=1 to 6 do
      interve[i]:=f1^.interve[i];
    2:for i:=1 to 6 do
      interve[i]:=f2^.interve[i];
    3:for i:=1 to 6 do
      interve[i]:=f3^.interve[i];
    end;
    gotoxy(0,3);
    write('INVERSION DES PARTIES:');
    gotoxy(0,4);
    write('----------------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> ');
    gotoxy(0,7);
    write('2-GRAPHE-------> ');
    gotoxy(0,9);
    write('3-ARBRE--------> ');
    gotoxy(0,11);
    write('4-LINEAIRE-----> ');
    gotoxy(0,13);
    write('5-MATH.+.------> ');
    gotoxy(0,15);
    write('6-MATH.-.------> ');
    x:=17;
    y:=3;
    for i:=1 to 6 do
    begin
        gotoxy(x,y+(2*i));
```

```
    if interve[i]=true then write('OUI')
                       else write('NON');
end;
ligne;
gotoxy(0,19);
write('SOIT A+B=C ,PEUT-ON INTERVERTIR LES ');
gotoxy(0,20);
write('LOCALISATIONS DE A ET DE B ?');
gotoxy(0,21);
write('POUR QUELLES REPRESENTATIONS ?');
gotoxy(0,22);
write('R:-');
gotoxy(0,23);
write('(ESC = ARRET DE LA MODIFICATION)');
gotoxy(2,22);
if keypress then read(keyboard,ch);
while not keypress do begin end;
read(keyboard,ch);
while ord(ch)<>32 do
begin
    if ord(ch)=27
    then begin
         fin:=true;
         exit(modifier);
         end

    else
         if (ord(ch)>=49) and
            (ord(ch)<=54)
         then begin
              write(ch);
              i:=ord(ch)-48;
              case i of
              1:gotoxy(17,5);
              2:gotoxy(17,7);
              3:gotoxy(17,9);
              4:gotoxy(17,11);
              5:gotoxy(17,13);
              6:gotoxy(17,15);
              end;
              if interve[i] = false
              then begin
                   interve[i]:=true;
                   write('OUI');
                   end
              else begin
                   interve[i]:=false;
                   write('NON');
                   end;
              case i of
              1:gotoxy(0,5);
              2:gotoxy(0,7);
              3:gotoxy(0,9);
              4:gotoxy(0,11);
              5:gotoxy(0,13);
              6:gotoxy(0,15);
              end;

         end;
read(keyboard,ch);
gotoxy(2,22);
write('-');
```

```
    gotoxy(2,22);

    end;
    smod4;
end;


procedure MOD6(f:nofich);


var lig,rep:string[9];
    nbrex:packed array[1..6,1..6] of numex;
    nbrex2:packed array[1..5] of numex;

procedure SUITE2;

begin
    effecr2;
    gotoxy(0,19);
    write('INTRODUISEZ UN NOMBRE D EXERCICES');
    gotoxy(0,20);
    write('COMPRIS ENTRE 00 ET 25');
    gotoxy(31,6+(nb*2));
    write('--');
    gotoxy(31,6+(nb*2));
    nc:=0;
    read(keyboard,ch);
    while (nc<2) or (not eoln(keyboard) ) do
    begin
        if (nc=0) and
           (ord(ch)>=48) and
           (ord(ch)<=50)

        then begin
             write(ch);
             dizaine:=(ord(ch)-48)*10;
             nc:=nc+1;
             end

        else if (nc=1) and
                (ord(ch)>=48) and
                (ord(ch)<=53) and
                (dizaine=20)
             then begin
                  write(ch);
                  unite:=ord(ch)-48;
                  nc:=nc+1;
                  end

             else if (nc=1) and
                     (dizaine<>20) and
                     (ord(ch)>=48) and
                     (ord(ch)<=57)
                  then begin
                       write(ch);
                       unite:=ord(ch)-48;
                       nc:=nc+1;
                       end
                   else if (nc>0) and
                           (ord(ch)=8)
```

```
                        then begin
                             gotoxy(30+nc,6+(nb*2));
                             write('-');
                             gotoxy(30+nc,6+(nb*2));
                             nc:=nc-1;
                             end;

        read(keyboard,ch);

    end;
    nbrex2[nb]:=dizaine+unite;
    effecr2;
    gotoxy(0,19);
    write('GENRE D ASSOCIATION ?');
    gotoxy(0,21);
    write('R:');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA MODIFICATION)');

end;


procedure C5ASS;



begin
    gotoxy(0,3);
    write('NOMBRE D EXERCICES:');
    gotoxy(0,4);
    write('--------------------');
    gotoxy(0,6);
    write('POUR LES SITUATIONS LIEES PAR:');
    gotoxy(0,8);
    write('1- ASSOCIATION 2-1-1 ---------> ');
    gotoxy(0,10);
    write('2- ASSOCIATION 2-2 -----------> ');
    gotoxy(0,12);
    write('3- ASSOCIATION 3-1 -----------> ');
    gotoxy(0,14);
    write('4- ASSOCIATION 4 -------------> ');
    gotoxy(0,16);
    write('5- AUCUNE ASSOCIATION --------> ');
    x:=31;
    y:=6;
    for i:=1 to 5 do
    begin
        gotoxy(x,y+(2*i));
        if nbrex2[i]<10 then write('0',nbrex2[i])
                        else write(nbrex2[i]);
    end;
    ligne;
    gotoxy(0,19);
    write('GENRE D ASSOCIATION ?');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA MODIFICATION)');
    gotoxy(2,21);
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
```

```
    while ord(ch)<>32 do
    begin
        if ord(ch)=27 then begin
                           fin:=true;
                           exit(modifier);
                           end
        else if (ord(ch)>=49) and
                (ord(ch)<=53)
             then begin
                  gotoxy(2,21);
                  write(ch);
                  nb:=ord(ch)-48;
                  gotoxy(0,6+(nb*2));
                  read(keyboard,ch);
                  while (ord(ch)>=49) and
                        (ord(ch)<=53) do
                  begin
                      gotoxy(2,21);
                      write(ch);
                      nb:=ord(ch)-48;
                      gotoxy(0,6+(nb*2));
                      read(keyboard,ch);
                  end;
                  if ord(ch)=32 then suite2 ;
                  end;
             gotoxy(2,21);
             write('-');
             gotoxy(2,21);
             read(keyboard,ch);
     end;
     for i:=1 to 5 do
     f3^.nbrex[i]:=nbrex2[i];
end;


procedure SUITE1;

begin
    effecr2;
    gotoxy(0,19);
    write('INTRODUISEZ UN NOMBRE D EXERCICES');
    gotoxy(0,20);
    write('COMPRIS ENTRE 00 ET 25');
    gotoxy(x+25,y+nb-1);
    write('--');
    gotoxy(x+25,y+nb-1);
    nc:=0;
    read(keyboard,ch);
    while (nc<2) or (not eoln(keyboard) ) do
    begin
        if (nc=0) and
           (ord(ch)>=48) and
           (ord(ch)<=50)

        then begin
             write(ch);
             dizaine:=(ord(ch)-48)*10;
             nc:=nc+1;
             end

        else if (nc=1) and
                (ord(ch)>=48) and
```

```
                (ord(ch)<=53) and
                (dizaine=20)

            then begin
                 write(ch);
                 unite:=ord(ch)-48;
                 nc:=nc+1;
                 end

            else if (nc=1) and
                    (dizaine<>20) and
                    (ord(ch)>=48) and
                    (ord(ch)<=57)
                 then begin
                      write(ch);
                      unite:=ord(ch)-48;
                      nc:=nc+1;
                      end

                 else if (nc>0) and
                         (ord(ch)=8)

                      then begin
                           gotoxy(x+24+nc,y+nb-1);
                           write('-');
                           gotoxy(x+24+nc,y+nb-1);
                           nc:=nc-1;
                           end;

        read(keyboard,ch);

    end;
    nbrex[i,nb]:=dizaine+unite;
    effecr2;
    gotoxy(0,19);
    write('NUMERO DE REPRESENTATION ?');
    gotoxy(0,21);
    write('R:');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA MODIFICATION)');

end;



procedure CSSEL1;

begin

    gotoxy(0,3);
    write('NOMBRE D EXERCICES :');
    gotoxy(0,4);
    write('--------------------');
    gotoxy(0,5);
    write('SOIT UNE SITUATION DE DEPART PRESENTEE');
    gotoxy(0,6);
    write('SOUS FORME DE ');
    gotoxy(0,8);
    write('ET UNE SITUATION A RECONNAITRE');
    gotoxy(0,9);
    write('SOUS FORME DE :                    NBR.EX.:');
```

```
    gotoxy(32,10);
    write('-------');
    x:=10;
    y:=11;
    for i:=1 to 6 do
    begin
        gotoxy(x,y+i-1);
        write(i,'-');
        case i of
        1:write('DIAGRAMME---------->');
        2:write('GRAPHE------------->');
        3:write('ARBRE-------------->');
        4:write('LINEAIRE----------->');
        5:write('MATH.+.------------>');
        6:write('MATH.-.------------>');
        end;
    end;
    ligne;
end;

procedure C5SEL2;

begin
    for i:=1 to 6 do
    begin
        case i of
        1:begin
          rep:='DIAGRAMME';
          lig:='----------';
          end;
        2:begin
          rep:='GRAPHE    ';
          lig:='------    ';
          end;
        3:begin
          rep:='ARBRE     ';
          lig:='-----     ';
          end;
        4:begin
          rep:='LINEAIRE  ';
          lig:='--------  ';
          end;
        5:begin
          rep:='MATH.+.   ';
          lig:='-------   ';
          end;
        6:begin
          rep:='MATH.-.   ';
          lig:='-------   ';
          end;
        end;
        gotoxy(14,6);
        write(rep);
        gotoxy(14,7);
        write(lig);
        x:=35;
        y:=10;
        for k:=1 to 6 do
        begin
        gotoxy(x,y+k);
        if nbrex[i,k]<10 then write('0',nbrex[i,k])
                         else write(nbrex[i,k]);
```

```
        end;
        gotoxy(0,19);
        write('NUMERO DE REPRESENTATION ?');
        gotoxy(0,21);
        write('R:-');
        gotoxy(0,23);
        write('(ESC = ARRET DE LA MODIFICATION)');
        gotoxy(2,21);
        x:=10;
        y:=11;
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);

        while ord(ch)<>32 do
        begin
            if ord(ch)=27 then begin
                                fin:=true;
                                exit(modifier);
                                end
            else if (ord(ch)>=49) and
                    (ord(ch)<=54)

                 then begin
                      nb:=ord(ch)-48;
                      write(ch);
                      gotoxy(x,y+nb-1);
                      read(keyboard,ch);
                      while (ord(ch)>=49) and
                            (ord(ch)<=54) do
                      begin
                          gotoxy(2,21);
                          write(ch);
                          nb:=ord(ch)-48;
                          gotoxy(x,y+nb-1);
                          read(keyboard,ch);
                      end;
                      if ord(ch)=32 then suite1;
                      end;
            gotoxy(2,21);
            write('-');
            gotoxy(2,21);
            read(keyboard,ch);
        end;

    end;
    case f of
    1:for i:=1 to 6 do
      for j:=1 to 6 do
      f1^.nbrex[i,j]:=nbrex[i,j];
    2:for i:=1 to 6 do
      for j:=1 to 6 do
      f2^.nbrex[i,j]:=nbrex[i,j];
    end;
end;

begin
    fin:=false;
    case f of
    1:begin
      for i:=1 to 6 do
      for j:=1 to 6 do
```

```
      nbrex[i,j]:=f1^.nbrex[i,j];
      c5sel1;
      c5sel2;
      end;
    2:begin
      for i:=1 to 6 do
      for j:=1 to 6 do
      nbrex[i,j]:=f2^.nbrex[i,j];
      c5sel1;
      c5sel2;
      end;
    3:begin
      for i:=1 to 5 do
      nbrex2[i]:=f3^.nbrex[i];
      c5ass;
      end;
    end;

end;

procedure MOD7(f:nofich) ;


var valmax:10..100;

begin
    fin:=false;
    case f of
    1:valmax:=f1^.valmax;
    2:valmax:=f2^.valmax;
    3:valmax:=f3^.valmax;
    end;
    gotoxy(0,3);
    write('VALEUR MAXIMALE DE LA SOMME');
    gotoxy(0,4);
    write('---------------------------');
    gotoxy(0,6);
    write('SOIT A+B=C. LA PLUS GRANDE VALEUR QUE');
    gotoxy(0,8);
    write('C PEUT ATTEINDRE EST ');
    if valmax=100 then write('100')
    else write('0',valmax);
    ligne;
    gotoxy(0,19);
    write('VALEUR MAXIMALE DE LA SOMME ?');
    gotoxy(0,20);
    write('COMPRISE ENTRE 10 ET 100');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA MODIFICATION)');
    x:=21;
    y:=8;
    gotoxy(x,y);
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    if ord(ch)<>32 then
    begin
        nb:=0;
        centaine:=0;
        dizaine:=0;
        unite:=0;
        write('---');
```

```
gotoxy(x,y);
while (nb<3) or
      (ord(ch)<>32) do
begin

if ord(ch)=27 then begin
                  fin:=true;
                  exit(modifier);
                  end
else if (nb=0) and
        ((ord(ch)=48) or (ord(ch)=49))
    then begin
         write(ch);
         nb:=nb+1;
         centaine:=ord(ch)-48;
         end
    else if (nb=1) and
            (centaine=1) and
            (ord(ch)=48)
         then begin
              write(ch);
              dizaine:=0;
              nb:=nb+1;
              end
         else if (nb=1) and
                 (centaine=0) and
                 (ord(ch)>=49) and
                 (ord(ch)<=57)
              then begin
                   write(ch);
                   dizaine:=ord(ch)-48;
                   nb:=nb+1;
                   end
              else if (nb=2) and
                   (centaine=1) and
                   (ord(ch)=48)
                   then begin
                        write(ch);
                        nb:=nb+1;
                        unite:=0;
                        end
                   else if (nb=2) and
                           (centaine=0) and
                           (ord(ch)>=48) and
                           (ord(ch)<=57)
                        then begin
                             write(ch);
                             nb:=nb+1;
                             unite:=ord(ch)-48;
                             end
                        else if (nb<=3) and
                                (nb>=1) and
                                (ord(ch)=8)
                             then begin
                                  nb:=nb-1;
                                  gotoxy(x+nb,y);
                                  write('-');
                                  gotoxy(x+nb,y);
                                  end;

read(keyboard,ch);
end;
```

```
        valmax:=(centaine*100)+
                (dizaine*10)+
                (unite);
    end;
    case f of
    1:f1^.valmax:=valmax;
    2:f2^.valmax:=valmax;
    3:f3^.valmax:=valmax;
    end;
end;
```

```
procedure MODS(f:nofich);

var vardess:packed array[1..2] of boolean;

begin
    case f of
    1:for i:=1 to 2 do
      vardess[i]:=f1^.vardess[i];
    2:for i:=1 to 2 do
      vardess[i]:=f2^.vardess[i];
    3:for i:=1 to 2 do
      vardess[i]:=f3^.vardess[i];
    end;
    gotoxy(0,3);
    write('DESSIN VARIABLE:');
    gotoxy(0,4);
    write('----------------');
    gotoxy(0,5);
    write('1-DIAGRAMME----> ');
    if vardess[1]=true then write('OUI')
                       else write('NON');

    gotoxy(0,7);
    write('2-GRAPHE-------> ');
    if vardess[2]=true then write('OUI')
                       else write('NON');
    ligne;
    gotoxy(0,19);
    write('UNE MEME REPRESENTATION PEUT-ELLE ETRE');
    gotoxy(0,20);
    write('DESSINEE DE DIFFERENTES MANIERES ?');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA MODIFICATION)');
    gotoxy(2,21);
    fin:=false;
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27
        then begin
             fin:=true;
             exit(modifier);
             end
        else
             if (ord(ch)>=49) and
                (ord(ch)<=50)
             then begin
                  write(ch);
                  i:=ord(ch)-48;
                  case i of
                  1:gotoxy(17,5);
                  2:gotoxy(17,7);
                  end;
                  if vardess[i] = false
                  then begin
```

```
                    vardess[i]:=true;
                    write('OUI');
                        end
                else begin
                    vardess[i]:=false;
                    write('NON');
                    end;
                case i of
                1:gotoxy(0,5);
                2:gotoxy(0,7);
                end;

            end;
    read(keyboard,ch);
    gotoxy(2,21);
    write('-');
    gotoxy(2,21);

    end;

    case f of
    1:for i:=1 to 2 do
      f1^.vardess[i]:=vardess[i];
    2:for i:=1 to 2 do
      f2^.vardess[i]:=vardess[i];
    3:for i:=1 to 2 do
      f3^.vardess[i]:=vardess[i];
    end;
end;



procedure MOD8(f:nofich);


var passdix:boolean;

begin
    case f of
    1:passdix:=f1^.passdix;
    2:passdix:=f2^.passdix;
    3:passdix:=f3^.passdix;
    end;

    fin:=false;
    gotoxy(0,3);
    write('REPORT A LA DIZAINE');
    gotoxy(0,4);
    write('--------------------');
    gotoxy(0,6);
    write('LES SITUATIONS OU LE CALCUL DE A+B');
    gotoxy(0,8);
    write('OCCASIONNENT UN REPORT A LA DIZAINE');
    gotoxy(0,10);
    write('SONT ADMISES ------> ');
    if passdix=true then write('OUI')
                    else write('NON');
    gotoxy(21,11);
    write('---');
    ligne;
    gotoxy(0,19);
```

```
    write('REPORT A LA DIZAINE ADMIS ?');
    gotoxy(0,20);
    write('        (O=OUI N=NON)');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA MODIFICATION)');
    gotoxy(2,21);
    if keypress then read(keyboard,ch);
    while not keypress do begin end;
    read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27 then begin
                           fin:=true;
                           exit(modifier);
                           end
        else if ord(ch)=79
             then begin
                  write(ch);
                  gotoxy(21,10);
                  write('OUI');
                  gotoxy(2,21);
                  passdix:=true;
                  end
             else if ord(ch)=78
                  then begin
                       write(ch);
                       gotoxy(21,10);
                       write('NON');
                       gotoxy(2,21);
                       passdix:=false;
                       end
                  else begin
                       write('-');
                       gotoxy(2,21);
                       end;
    read(keyboard,ch);

    end;
    case f of
    1:f1^.passdix:=passdix;
    2:f2^.passdix:=passdix;
    3:f3^.passdix:=passdix;
    end;

end;

procedure MOD9 (f:nofich);


var tabsit1:packed array[1..9]of boolean;
    ok:boolean;

procedure DIS1;
begin
    gotoxy(0,6);
    write('-MEMES SYMBOLES:');
    gotoxy(5,8);
    write('1- A+?=B , ?+B=A ---------> ');
    if tabsit1[1]=true then begin
                            write('OUI');
```

```
                              nc:=nc+1;
                              end
                         else write('NON');

    gotoxy(5,9);
    write('    (B>A)    (A>B)');
    gotoxy(0,11);
    write('-A OU B REMPLACE PAR C:');
    gotoxy(5,13);
    write('2- C+B=? , A+C=? ----------> ');
    if tabsit1[2]=true then begin
                              write('OUI');
                              nc:=nc+1;
                              end
                         else write('NON');
    gotoxy(5,15);
    write('3- ?+B=C , A+?=C ----------> ');
    if tabsit1[3]=true then begin
                              write('OUI');
                              nc:=nc+1;
                              end
                         else write('NON');
end;

procedure DIS2;
begin
    gotoxy(0,6);
    write('-A OU B REMPLACE PAR X:');
    gotoxy(5,8);
    write('4- X+B=? , A+X=? ----------> ');
    if tabsit1[4]=true then begin
                              write('OUI');
                              nc:=nc+1;
                              end
                         else write('NON');
    gotoxy(5,10);
    write('5- ?+B=X , A+?=X !');
    gotoxy(5,11);
    write('    (X>B)    (X>A) !');
    gotoxy(5,12);
    write('                   !---------> ');
    if tabsit1[5]=true then begin
                              write('OUI');
                              nc:=nc+1;
                              end
                         else write('NON');
    gotoxy(5,13);
    write('    ?+X=B , X+?=A !');
    gotoxy(5,14);
    write('    (X<B)    (X<A) !');
end;

procedure DIS3;
begin
    gotoxy(0,6);
    write('-A ET B REMPLACES PAR C ET X:');
    gotoxy(5,8);
    write('6- X+C=? ------------------> ');
    if tabsit1[6]=true then begin
                              write('OUI');
                              nc:=nc+1;
                              end
```

```
                        else write('NON');
    gotoxy(5,10);
    write('7- ?+C=X , X+?=C ----------> ');
    if tabsit1[7]=true then begin
                            write('OUI');
                            nc:=nc+1;
                            end
                        else write('NON');
    gotoxy(5,11);
    write('    (X>C)    (C>X) ');
    gotoxy(0,13);
    write('-A ET B REMPLACES PAR X ET Y (X>Y):');
    gotoxy(5,15);
    write('8- X+Y=? ------------------> ');
    if tabsit1[8]=true then begin
                            write('OUI');
                            nc:=nc+1;
                            end
                        else write('NON');
    gotoxy(5,17);
    write('9- ?+Y=X ------------------> ');
    if tabsit1[9]=true then begin
                            write('OUI');
                            nc:=nc+1;
                            end
                        else write('NON');
end;


procedure DC8;

begin
    fin:=false;
    nc:=0;
    gotoxy(0,3);
    write('DISTORSIONS PAR RAPPORT A A+B=? SI A+B=C');
    gotoxy(0,4);
    write('--------------------------------');
    dis1;
    ligne;
    gotoxy(0,19);
    write('SELECTIONNEZ LES DISTORSIONS ADMISES');
    gotoxy(0,20);
    write('        (AU MOINS 2 PARMI 9 AU TOTAL)');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA MODIFICATION)');
    gotoxy(2,21);
    ok:=false;
end;

begin
    dc8;
    while ok=false do
    begin
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27 then begin
```

```
                        fin:=true;
                        exit(modifier);
                        end
    else if (ord(ch)>=49) and
            (ord(ch)<=51)

         then begin
              nb:=ord(ch)-48;
              gotoxy(2,21);
              write(ch);
              case nb of
              1:gotoxy(33,8);
              2:gotoxy(33,13);
              3:gotoxy(33,15);
              end;
              if tabsit1[nb]=false
              then begin
                   tabsit1[nb]:=true;
                   write('OUI');
                   nc:=nc+1;
                   end
              else begin
                   tabsit1[nb]:=false;
                   write('NON');
                   nc:=nc-1;
                   end;
              case nb of
              1:gotoxy(5,8);
              2:gotoxy(5,13);
              3:gotoxy(5,15);
              end;
              end
         else begin
              gotoxy(2,21);
              write('-');
              gotoxy(2,21);
              end;

    read(keyboard,ch);
end;
    effecr3;
    dis2;
    gotoxy(2,21);
    write('-');
    gotoxy(2,21);
    read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27 then begin
                           fin:=true;
                           exit(modifier);
                           end
        else if (ord(ch)>=52) and
                (ord(ch)<=53)

             then begin
                  nb:=ord(ch)-48;
                  gotoxy(2,21);
                  write(ch);
                  case nb of
                  4:gotoxy(33,8);
                  5:gotoxy(33,12);
```

```
                end;
                if tabsit1[nb]=false
                then begin
                     tabsit1[nb]:=true;
                     write('OUI');
                     nc:=nc+1;
                     end
                else begin
                     tabsit1[nb]:=false;
                     write('NON');
                     nc:=nc-1;
                     end;
                case nb of
                4:gotoxy(5,8);
                5:gotoxy(5,10);
                end;
                end

           else begin
                gotoxy(2,21);
                write('-');
                gotoxy(2,21);
                end;

    read(keyboard,ch);
end;

effecr3;
dis3;
gotoxy(2,21);
write('-');
gotoxy(2,21);
read(keyboard,ch);
while ord(ch)<>32 do
begin
    if ord(ch)=27 then begin
                       fin:=true;
                       exit(modifier);
                       end
    else if (ord(ch)>=54) and
            (ord(ch)<=57)

         then begin
              nb:=ord(ch)-48;
              gotoxy(2,21);
              write(ch);
              case nb of
              6:gotoxy(33,8);
              7:gotoxy(33,10);
              8:gotoxy(33,15);
              9:gotoxy(33,17);
              end;
              if tabsit1[nb]=false
              then begin
                   tabsit1[nb]:=true;
                   write('OUI');
                   nc:=nc+1;
                   end
              else begin
                   tabsit1[nb]:=false;
                   write('NON');
                   nc:=nc-1;
```

```
                        end;
                    case nb of
                    6:gotoxy(5,8);
                    7:gotoxy(5,10);
                    8:gotoxy(5,15);
                    9:gotoxy(5,17);
                    end;
                    end
                else begin
                    gotoxy(2,21);
                    write('-');
                    gotoxy(2,21);
                    end;

          read(keyboard,ch);
        end;

        if nc<2 then begin
                        effecr3;
                        dis1;
                        gotoxy(2,21);
                        write('-');
                        gotoxy(2,21);
                        end
                else ok:=true;
    end;
    case f of
    1:for i:=1 to 9 do
      f1^.tabsit1[i]:=tabsit1[i];
    2:for i:=1 to 9 do
      f2^.tabsit1[i]:=tabsit1[i];
    3:for i:=1 to 9 do
      f3^.tabsit1[i]:=tabsit1[i];
    end;
end;


procedure MOD10 (f:nofich);


var
    tabsit2:packed array[1..9]of boolean;
    var ok:boolean;

procedure DIS1;
begin
    gotoxy(0,6);
    write('-MEMES SYMBOLES:');
    gotoxy(5,8);
    write('1- A+C=? ----------------> ');
    if tabsit2[1]=true then begin
                            write('OUI');
                            nc:=nc+1;
                            end
                       else write('NON');
    gotoxy(0,10);
    write('-A OU C REMPLACE PAR B:');
    gotoxy(5,12);
    write('2- C+B=? , A+B=? ----------> ');
    if tabsit2[2]=true then begin
                            write('OUI');
```

```
                                nc:=nc+1;
                                end
                            else write('NON');
    gotoxy(5,14);
    write('3- ?+B=C , A+?=B , ?+B=A -> ');
    if tabsit2[3]=true then begin
                                write('OUI');
                                nc:=nc+1;
                                end
                            else write('NON');
    gotoxy(5,15);
    write('              (B>A)    (A>B)');

end;

procedure DIS2;
begin
    gotoxy(0,6);
    write('-A OU C REMPLACE PAR X:');
    gotoxy(5,8);
    write('4- X+C=? , A+X=? ----------> ');
    if tabsit2[4]=true then begin
                                write('OUI');
                                nc:=nc+1;
                                end
                            else write('NON');
    gotoxy(5,10);
    write('5- ?+C=X , A+?=X !');
    gotoxy(5,11);
    write('    (X>C)    (X>A) !');
    gotoxy(5,12);
    write('                     !---------> ');
    if tabsit2[5]=true then begin
                                write('OUI');
                                nc:=nc+1;
                                end
                            else write('NON');
    gotoxy(5,13);
    write('    ?+X=C , X+?=A !');
    gotoxy(5,14);
    write('    (X<C)    (X<A) !');
end;

procedure DIS3;
begin
    gotoxy(0,6);
    write('-A ET C REMPLACES PAR B ET X:');
    gotoxy(5,8);
    write('6- X+B=? ------------------> ');
    if tabsit2[6]=true then begin
                                write('OUI');
                                nc:=nc+1;
                                end
                            else write('NON');
    gotoxy(5,10);
    write('7- ?+B=X , X+?=B ----------> ');
    if tabsit2[7]=true then begin
                                write('OUI');
                                nc:=nc+1;
                                end
                            else write('NON');
    gotoxy(5,11);
```

```
    write('    (X>B)    (B>X) ');
    gotoxy(0,13);
    write('-A ET C REMPLACES PAR X ET Y (X>Y):');
    gotoxy(5,15);
    write('8- X+Y=? ------------------> ');
    if tabsit2[8]=true then begin
                            write('OUI');
                            nc:=nc+1;
                            end
                       else write('NON');
    gotoxy(5,17);
    write('9- ?+Y=X ------------------> ');
    if tabsit2[9]=true then begin
                            write('OUI');
                            nc:=nc+1;
                            end
                       else write('NON');
end;

procedure DC9;

begin
    fin:=false;
    nc:=0;
    gotoxy(0,3);
    write('DISTORSIONS PAR RAPPORT A A+?=C SI A+B=C');
    gotoxy(0,4);
    write('-----------------------------------');
    dis1;
    ligne;
    gotoxy(0,19);
    write('SELECTIONNEZ LES DISTORSIONS ADMISES');
    gotoxy(0,20);
    write('        (AU MOINS 2 PARMI 9 AU TOTAL)');
    gotoxy(0,21);
    write('R:-');
    gotoxy(0,23);
    write('(ESC = ARRET DE LA MODIFICATION)');
    gotoxy(2,21);
    ok:=false;
end;

begin
    dc9;
    while ok=false do
    begin
        if keypress then read(keyboard,ch);
        while not keypress do begin end;
        read(keyboard,ch);
    while ord(ch)<>32 do
    begin
        if ord(ch)=27 then begin
                           fin:=true;
                           exit(modifier);
                           end
        else if (ord(ch)>=49) and
                (ord(ch)<=51)

             then begin
                  nb:=ord(ch)-48;
                  gotoxy(2,21);
```

```
                write(ch);
                case nb of
                1:gotoxy(33,8);
                2:gotoxy(33,12);
                3:gotoxy(33,14);
                end;
                if tabsit2[nb]=false
                then begin
                     tabsit2[nb]:=true;
                     write('OUI');
                     nc:=nc+1;
                     end
                else begin
                     tabsit2[nb]:=false;
                     write('NON');
                     nc:=nc-1;
                     end;
                case nb of
                1:gotoxy(5,8);
                2:gotoxy(5,12);
                3:gotoxy(5,14);
                end;
                end
           else begin
                gotoxy(2,21);
                write('-');
                gotoxy(2,21);
                end;

     read(keyboard,ch);
end;
     effecr3;
     dis2;
     gotoxy(2,21);
     write('-');
     gotoxy(2,21);
     read(keyboard,ch);
     while ord(ch)<>32 do
     begin
          if ord(ch)=27 then begin
                             fin:=true;
                             exit(modifier);
                             end
          else if (ord(ch)>=52) and
                  (ord(ch)<=53)

               then begin
                    nb:=ord(ch)-48;
                    gotoxy(2,21);
                    write(ch);
                    case nb of
                    4:gotoxy(33,8);
                    5:gotoxy(33,12);
                    end;
                    if tabsit2[nb]=false
                    then begin
                         tabsit2[nb]:=true;
                         write('OUI');
                         nc:=nc+1;
                         end
                    else begin
                         tabsit2[nb]:=false;
```

```
                    write('NON');
                    nc:=nc-1;
                    end;
                case nb of
                4:gotoxy(5,8);
                5:gotoxy(5,10);
                end;
                end

          else begin
                gotoxy(2,21);
                write('-');
                gotoxy(2,21);
                end;

    read(keyboard,ch);
end;

effecr3;
dis3;
gotoxy(2,21);
write('-');
gotoxy(2,21);
read(keyboard,ch);
while ord(ch)<>32 do
begin
    if ord(ch)=27 then begin
                       fin:=true;
                       exit(modifier);
                       end
    else if (ord(ch)>=54) and
            (ord(ch)<=57)

          then begin
                nb:=ord(ch)-48;
                gotoxy(2,21);
                write(ch);
                case nb of
                6:gotoxy(33,8);
                7:gotoxy(33,10);
                8:gotoxy(33,15);
                9:gotoxy(33,17);
                end;
                if tabsit2[nb]=false
                then begin
                     tabsit2[nb]:=true;
                     write('OUI');
                     nc:=nc+1;
                     end
                else begin
                     tabsit2[nb]:=false;
                     write('NON');
                     nc:=nc-1;
                     end;
                case nb of
                6:gotoxy(5,8);
                7:gotoxy(5,10);
                8:gotoxy(5,15);
                9:gotoxy(5,17);
                end;
                end
          else begin
```

```
                        gotoxy(2,21);
                        write('-');
                        gotoxy(2,21);
                        end;

            read(keyboard,ch);
        end;

        if nc<2 then begin
                        effecr3;
                        dis1;
                        gotoxy(2,21);
                        write('-');
                        gotoxy(2,21);
                        end
                else ok:=true;
    end;
    case f of
    1:for i:=1 to 9 do
      f1^.tabsit2[i]:=tabsit2[i];
    2:for i:=1 to 9 do
      f2^.tabsit2[i]:=tabsit2[i];
    3:for i:=1 to 9 do
      f3^.tabsit2[i]:=tabsit2[i];
    end;
end;

begin
    effecr;
    mod1(f);
    effecr1;
    mod2(f);
    effecr1;
    mod3(f);
    effecr1;
    mod4(f);
    effecr1;
    mod5(f);
    effecr1;
    mod6(f);
    effecr1;
    mod7(f);
    effecr1;
    mod8(f);
    effecr1;
    mod9(f);
    effecr1;
    mod10(f);

end;
```